微信电子健康卡开放平台

实施对接方案

V 2.1

目录

第1章 整体架构

微信电子健康卡开放平台（后称开放平台），是针对电子健康卡项目对外开放的基础平台，医院服务号开发商（即 ISV）可以通过开放平台实现新用户建卡、跨院用户一键注册健康卡、老用户批量升级健康卡、用卡数据监测、统一卡包等功能。

开放平台与卡管平台交互，帮助卡管平台对电子健康卡进行创建和管理，同时开放平台会对 ISV 的请求提供鉴权、加密、审计等服务。

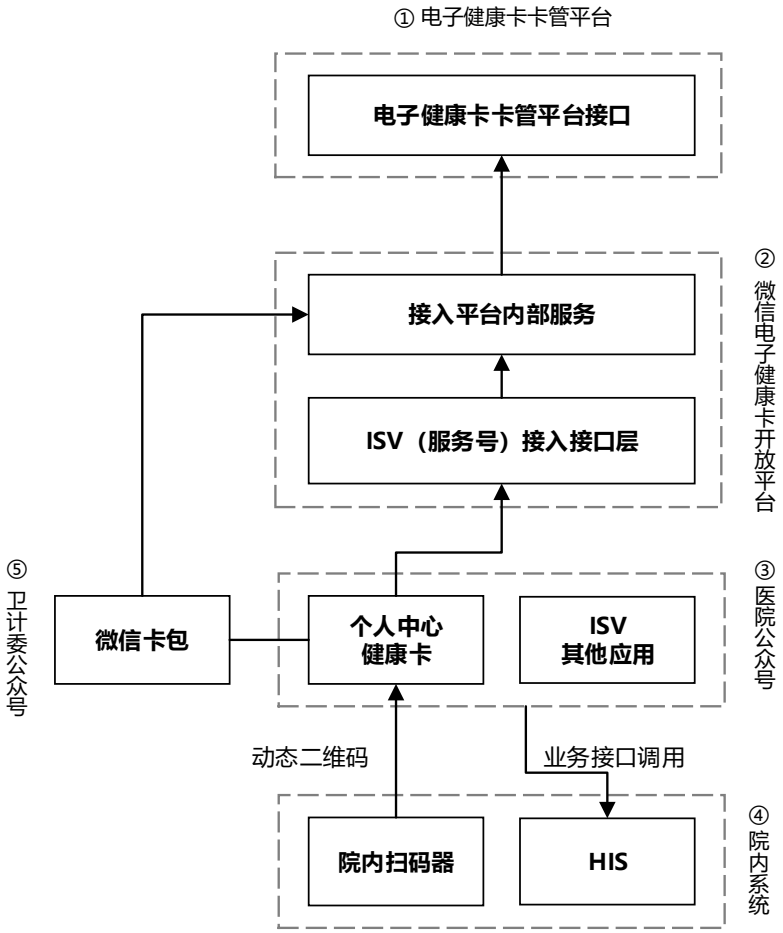

图 1.1 整体架构

微信电子健康卡开放平台全流程涉及以下几部分工作：

（1）电子健康卡卡管平台，负责创建、发放和管理健康卡；

（2）微信电子健康卡开放平台，负责协助医院完成微信渠道电子健康卡的

接入实施，提供标准实施方案，规范线上线下发卡、用卡流程，并协助验收实施效果；

（3）医院微信公众号、卡包，通过与微信电子健康卡开放平台对接，实现电子健康卡的线上申领、展示、扫与被扫等功能。

（4）医院院内系统，主要涉及院内扫码设备和 HIS 系统的相关改造，满足电子健康卡的建卡与识读等。

第2章 对接实施方案

2.1 医院入驻流程

微信电子健康卡开放平台（下称开放平台），是针对电子健康卡项目对外开放的基础平台，上接卡管平台，帮助卡管平台对电子健康卡进行创建和管理，同时开放平台会对医院服务号开发商的请求提供鉴权、加密、审计等服务。医院服务号开发商（即 ISV）可以通过开放平台实现新用户建卡、跨院用户一键注册健康卡、老用户批量升级健康卡、用卡数据监测、统一卡包等功能。

2.1.1 入驻流程

医院开发商通过官网注册入驻开放平台，通过提交营业执照、邀请医院入驻等步骤，获取开发授权凭证、医院 ID，通过对接开放平台提供的能力接口，快速实现电子健康卡的实名申领以及使用。

图 2.1 微信电子健康卡开放平台官网

具体操作步骤如下：

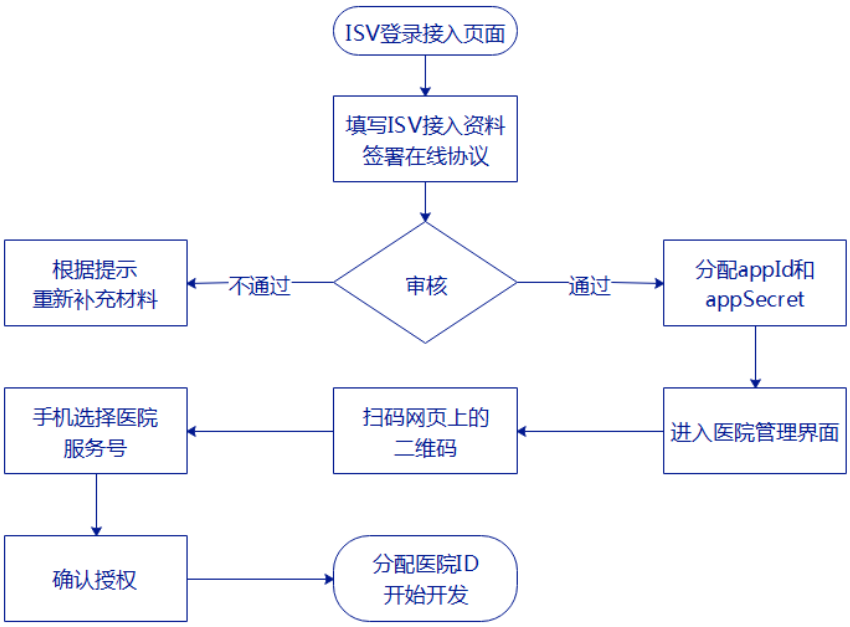

图 2.2 接入流程图

- 开发商登录开放平台官网（https://open.tengmed.com）注册并提交资料，审核认证资质。
- 审核通过后开发商在线签署协议，开放平台生成授权凭证 appId 和 appSecret，开发商获得授权。
- 开发商进入“医院入驻”菜单，将页面上的二维码发送医院的服务号创建者扫描。
- 创建者选择名下的医院服务号进行授权入驻，即绑定开发商与医院服务号的关联关系。绑定完成后，开放平台分配医院唯一标识医院 Id(hospital Id)。
- 开发商保存接口调用凭证：appId、appSecret 和 hospitalId，通过阅读官网开发者文档，开始对接微信电子健康卡开放平台。

2.1.2 入驻审核

2.1.2.1 开发商入驻审核

医院服务号开发商负责服务号的各项功能的开发，入驻微信电子健康卡开放平台流程为：开发商先在官网注册，注册完成后填写企业信息提交审核，填写企业信息包括企业名称、营业执照、联系人姓名、联系人邮箱、联系电话等。开放平台审核完成后，会以短信和邮件形式通知开发商，开发商登陆官网签署《健康卡开放平台开发者服务协议》，开放平台分配授权凭证 appId 和 appSecret。

图 2.3 开发商接入页面

2.1.2.2 医院入驻审核

医院开发商入驻完成后，需要邀请医院入驻，即绑定开发商与医院服务号的关联关系。

开发商入驻完成后，进入“医院入驻”菜单，将页面上的二维码发送给医院服务号创建者扫描。创建者选择名下的服务号进行授权入驻，授予的权限包括模板消息权限。

图 2.4 医院服务号授权流程

2.1.2.3 关联唯一医院 ID

医院 ID 是经服务号创建者授权开发商后，由开放平台分配的唯一 ID，它标识一个医院，它与开发商对应的 appId 绑定在一起，需要注意的是不同的开发商绑定同一个服务号，生成的是同一个医院 ID。医院 ID 也是开发商调用接口必备的公共入参。

服务号授权后，开发商再次进入“医院入驻”菜单，在页面点击“激活医院 ID”，填写医院信息提交激活申请。微信电子健康卡开放平台通过微信开放平台核实医院服务号身份，完成身份审核。

审核通过后，开放平台给开发商分配唯一的医院 ID（hospitalId）。

图 2.5 医院入驻页面

2.2 开发流程

服务号开发商入驻开放平台完成后，进入“开发配置”菜单，添加服务器的 IP 白名单和域名白名单，即向开放平台登记服务号服务器的 IP 和域名。

图 2.6 开发配置页面

开发商现在可以开始服务号的电子健康卡接入工作了，通过阅读开发者文档来熟悉对接流程和接口文档，使用 appId、appSecret 和 hospitalId 即可调用开放平台接口，使用开放平台提供的页面素材、页面源代码、接口测试 SDK、专业的对接技术支持可提高开发速度。

开发商一般需要对接开放平台的五个流程，包括新用户建卡流程、跨院用户一键注册健康卡流程、老用户批量升级流程、用卡数据监测流程、统一卡包流程、扫码就医流程。

注意：扫码入驻七天后仍未上线的医院，限制每个接口调用次数，**具体为 100 次/天/接口**，上线后自动解除限制。

2.3 新用户建卡流程（二卡合一）

从未申领过健康卡的用户，可在区域内任一医院上，通过填写身份证件信息（或上传身份证件）、联系方式等，经实名验证后注册健康卡，同时绑定院内就诊卡 ID。

健康卡的注册流程需完全替换原有就诊卡办理流程，实现二卡合一，不可同时存在电子健康卡和电子就诊卡。

核身验证：用户使用身份证信息注册时，开发商必须调用核身接口，确保实名注册。提交身份信息有两种方式，第一种是用户手动输入，在注册页面填写姓名、身份证号码、民族等；第二种是上传身份证照片，开放平台提供 OCR 接口，用于将图片识别为文本信息。

绑定限制：微信服务号、小程序可限制最多绑定 5 张，以方便用户为亲友注册健康卡。

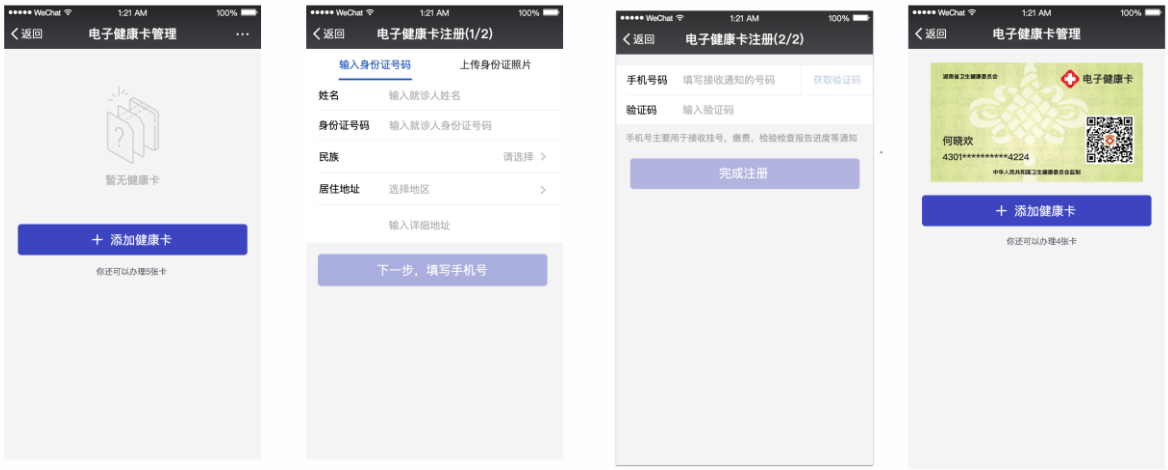

图 2.7 新用户注册流程页面示例图

2.3.1 业务流程

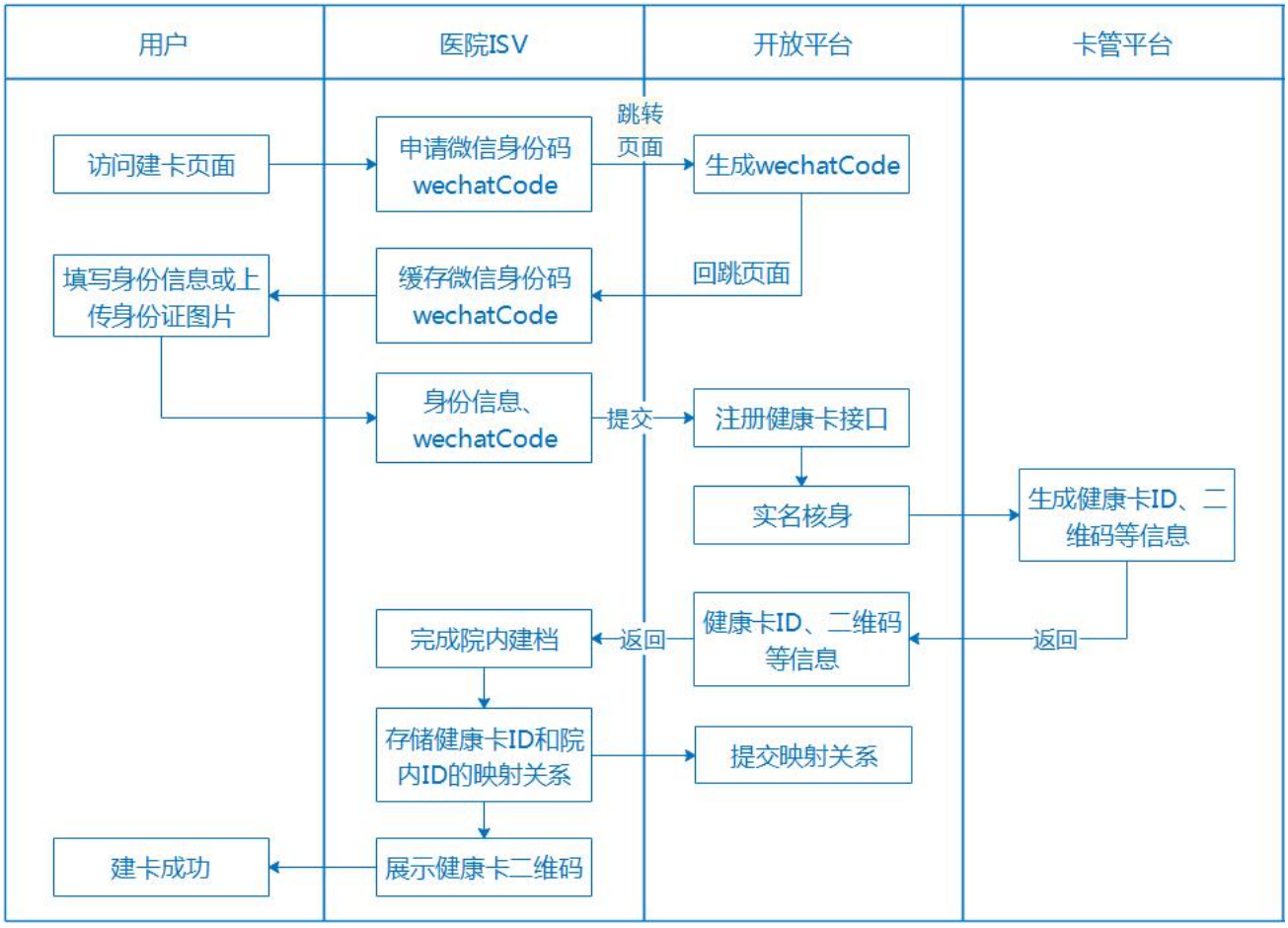

图 2.8 新用户注册流程

2.3.2 流程说明

1、用户访问建卡页面，ISV 跳转到开放平台页面申请获取用户微信身份码

wechatCode，用于在开放平台建卡；

2、用户在建卡页面输入身份信息，如姓名、身份证号码等；或者用户上传身份证照片，开放平台转换为身份证文本信息，ISV 需要调用 OCR 接口 进行转换；

3、ISV 获取用户信息，调用开放平台 注册健康卡接口，提交用户身份信息、wechatCode 参数，开放平台会对身份证号码和姓名进行实名核身；

4、核身通过后，开放平台先到卡管查询是否已注册，如果已注册，则直接返回健康卡 ID 和二维码等信息；如果没有，则由卡管平台完成注册，生成健康卡 ID 和二维码等信息，然后将其返回给开放平台，开放平台再返回给 ISV；

5、ISV 接收到开放平台响应后（表名核身通过且电子健康卡注册完成），先使用第 3 步获取的用户信息，到 HIS 查询用户是否已建档，如果已建档，则直接获取院内 ID；如果没有，则在 HIS 系统中新建电子就诊卡，然后再获取院内 ID；

6、ISV 存储健康卡 ID 与院内 ID 的映射关系，同时调用开放平台 绑定健康卡和院内 ID 关系接口 提交映射关系；如 HIS 已改造以健康卡 ID 为辅索引则无需关联；开放平台保存 ISV 提交的最近一次的映射关系，获取映射关系，请调用通过健康卡授权码获取健康卡接口 ；

7、建卡成功后，ISV 在页面展示二维码，用户进行后续操作。

2.3.3 业务规则

1、必须是有身份证的用户才能建立健康卡。

2、支持本人及家属健康卡。

开发平台向医院提供实名认证用户的姓名、身份证等信息，为保证用户数据安全，将用户的姓名、身份证等信息进行加密传输。

2.3.4 相关接口

- **注册健康卡接口**
- **OCR 接口**
- 绑定健康卡和院内 ID 关系接口
- 通过健康卡授权码获取健康卡接口

2.3.5 相关跳转 URL

- **获取用户微信身份码 wechatCode**

https://health.tengmed.com/open/getUserCode?redirect_uri=${redirect_uri}

2.4 跨院用户一键注册建卡流程

已注册过健康卡的用户，在其他服务号通过一键授权即可快速完成注册，无需重复输入身份信息。

对于有多个院内 ID 的用户，ISV 可在后台绑定用户最近使用或最近办理的一个院内 ID，或增加让用户自行选择绑定院内 ID 的环节；

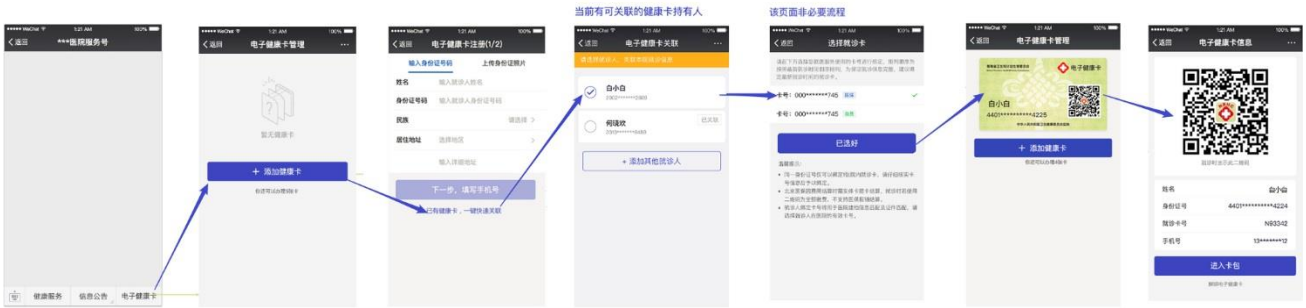

图 2.9 跨院用户一键注册健康卡示例图

2.4.1 业务流程

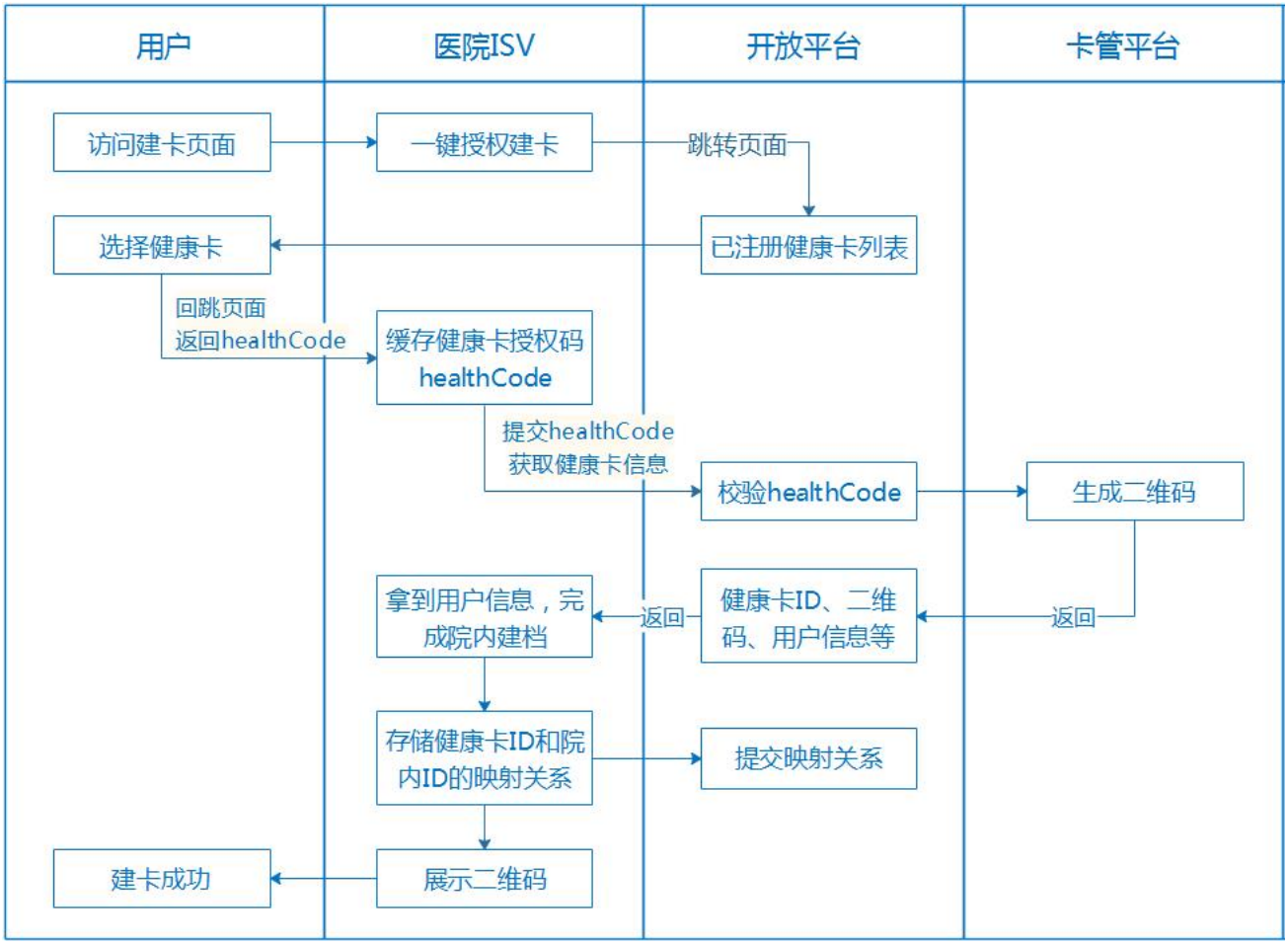

图 2.10 跨院一键注册建卡

2.4.2 流程说明

1、用户访问建卡页面，点击一键授权建卡，ISV 跳转到开放平台，开放平台显示用户在服务号注册过的健康卡列表，用户选择一张健康卡，开放平台回跳页面，并在路径中回传 healthCode；

2、ISV 缓存健康卡授权码 healthCode，将 healthCode 作为入参调用 通过健康卡授权码获取健康卡信息接口 ；

3、开放平台校验 healthCode 通过后，由卡管生成二维码并返回给开放平台，开放平台再将用户信息、健康卡 ID、二维码等返回 ISV；

4、ISV 获取用户信息，先到 HIS 查询用户是否已建档，如果已建档，则直接获取院内 ID；如果没有，则在 HIS 系统中新建电子就诊卡，然后再获取院内 ID；

5、ISV 存储健康卡 ID 与院内 ID 的映射关系，并调用 绑定健康卡和院内 ID 关系接口 提交映射关系；

6、跨院建卡成功，ISV 展示健康卡二维码。

2.4.3 业务规则

1、用户首次访问时，需要授权才能访问健康卡列表页。

2.4.4 相关接口

- 通过健康卡授权码获取健康卡信息接口
- 绑定健康卡和院内 ID 关系接口

2.4.5 相关跳转 URL

- 获取健康卡授权码 healthCode

  https://health.tengmed.com/open/getHealthCardList?redirect_uri=${redirect_uri}

2.5 老用户升级健康卡流程（批量升级，二卡合一）

针对已办理过院内就诊卡的历史患者数据，可通过批量核身、批量注册健康卡、批量通知用户领卡，实现就诊卡快速升级改造。用户只需点击健康卡升级的通知，或点击服务号健康卡入口，即可自动完成升级。

图 2.11 批量升级流程页面示例图

2.5.1 业务流程

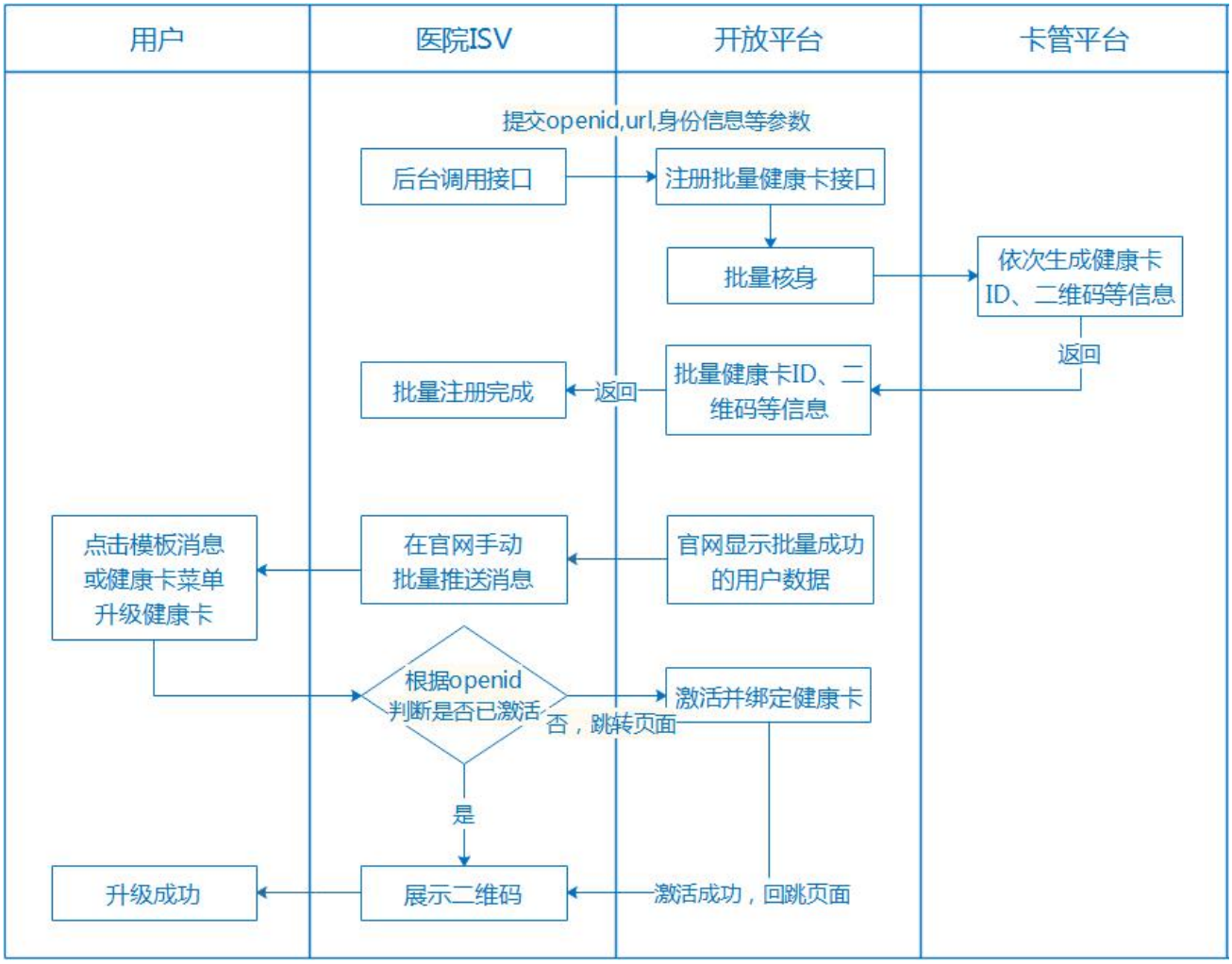

图 2.12 批量升级建卡

2.5.2 流程说明

1、ISV 后台调用 注册批量健康卡接口 发起批量注册操作，提交 openId、下发的模版消息 URL、身份信息等参数，开放平台对身份证号码和姓名进行批量实名核身；

2、核身通过后，开放平台先到卡管查询是否已注册，如果已注册，则直接返回健康卡 ID 和二维码等信息；如果没有，则由卡管平台完成注册，生成健康卡 ID 和二维码等信息，然后将其返回给开放平台；

3、所有用户注册完成后，开放平台将批量健康卡 ID、二维码等信息返回给 ISV；

4、ISV 登陆官网查看批量注册成功的用户 openId，点击一键领卡，给该批

次的所有用户推送升级健康卡的模板消息；

5、用户点击 模板消息 或者 电子健康卡菜单 升级健康卡(见本页老用户升级健康卡示例图)，ISV 根据 openId 判断用户是否已激活健康卡；

6、如果用户没有激活，则跳转到 开放平台激活页面 URL，由开放平台激活健康卡、绑定 openid 和健康卡 ID ；

7、开放平台激活成功后，自动回跳到 ISV 页面；

8、升级成功，进入健康卡列表页面。

2.5.3 业务规则

（1）使用模板消息激活时，需要满足公众号类型是服务号，且开通模板消息功能。订阅号不具备模板下发能力，可以采用菜单栏入口激活方式。

（2）模板消息的跳转页面需首先判断用户是否已批量激活过健康卡，如果没有则跳转至激活链接，激活之后返回至回跳链接。

2.5.4 相关接口

- 注册批量健康卡接口

2.5.5 相关跳转 URL 链接

- 跳转开放平台激活 URL

https://health.tengmed.com/open/batchActiveCard?open_id=${encryptData}&redirect_uri=${redirect_uri}

2.6 统一卡包流程

普通的微信卡包功能，发放卡券和领取卡券需为同一个服务号。统一卡包可实现由卫计委服务号统一发放健康卡，用户可通过区域内任意多家医院服务号将健康卡领取到卡包，并使用卡包中的扫码就医、预约挂号、健康档案等功能。用卡路径更短，跨院用卡更方便。

（1）领取卡包。用户注册健康卡后通过下发模板消息，提醒用户查看健康卡并领取到卡包。

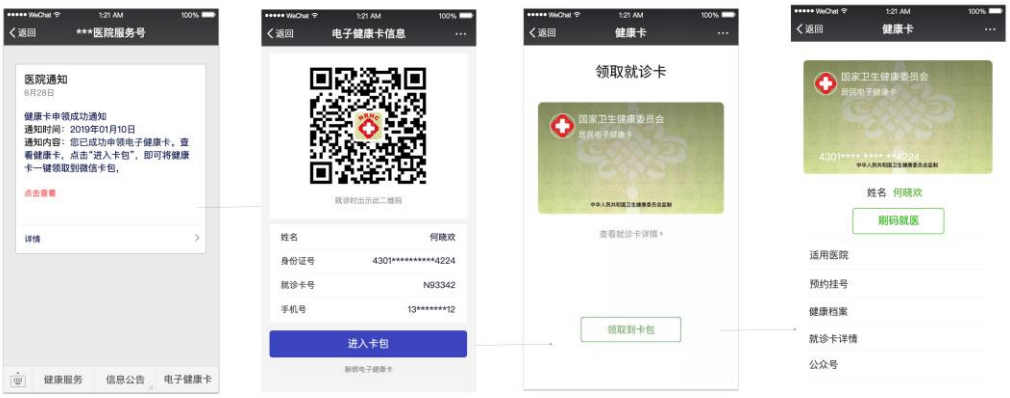

图 2.13 领取卡包流程示例图

（2）使用卡包。卡包中提供扫码就医、适用医院、预约挂号、健康档案等功能，满足线上线下的就医需求。

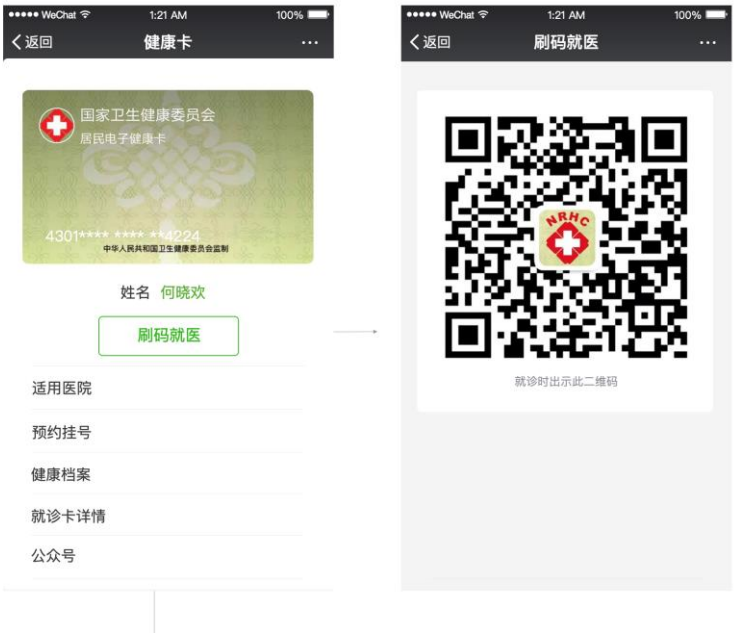

图 2.84 卡包功能示例图

2.6.1 流程说明

1、医院服务号接入流程：

注意：医院服务号必须在健康卡开放平台完成卡包-领卡配置，见下图：

图 2.17 卡包-领卡配置图

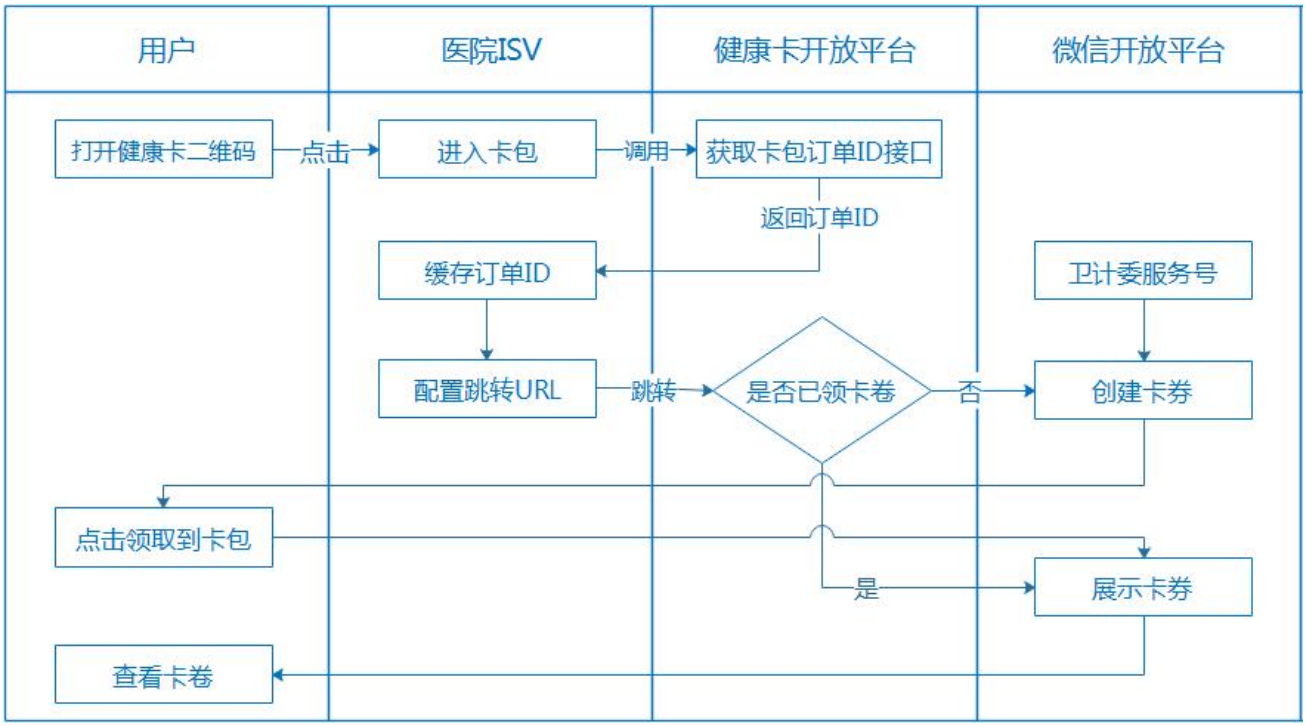

图 2.18 医院服务号（领卡服务号）接入流程图

1)用户进入服务号打开健康卡二维码，点击二维码下方的进入卡包 Button；

2)ISV 调用 获取卡包订单 ID 接口，获取并缓存订单 ID(orderId)；

3)ISV 使用 orderId 配置跳转领取卡卷的 URL；

4)开放平台根据 orderId 判断用户是否已领卡卷，如果没有，则由卫计委服务号创建，用户点击领取；如果已领，则直接展示；

5)用户在微信卡包中查看健康卡。

2.6.2 业务规则

1)可在卡包上配置挂号、微官网主页等 H5 链接，提供用户更好的使用体验。

2.7 业务协同与监测流程

电子健康卡使用时产生的的用卡数据，利用先进的数据仓库技术、报表技术以及成熟的报表工具按需建设健康卡用卡统计分析应用，满足健康卡用卡相关的固定报表或定制报表的统计分析需求，为辅助领导决策，业务优化，实现监管提供数据基础。

2.7.1 线上用卡数据监测业务流程

卡管平台提供前置接口程序，运用信息化手段进行在线监测。对微信渠道的电子健康卡建设情况和数据质量进行监测。通过调用接口的方式，可以满足业务监测需求。

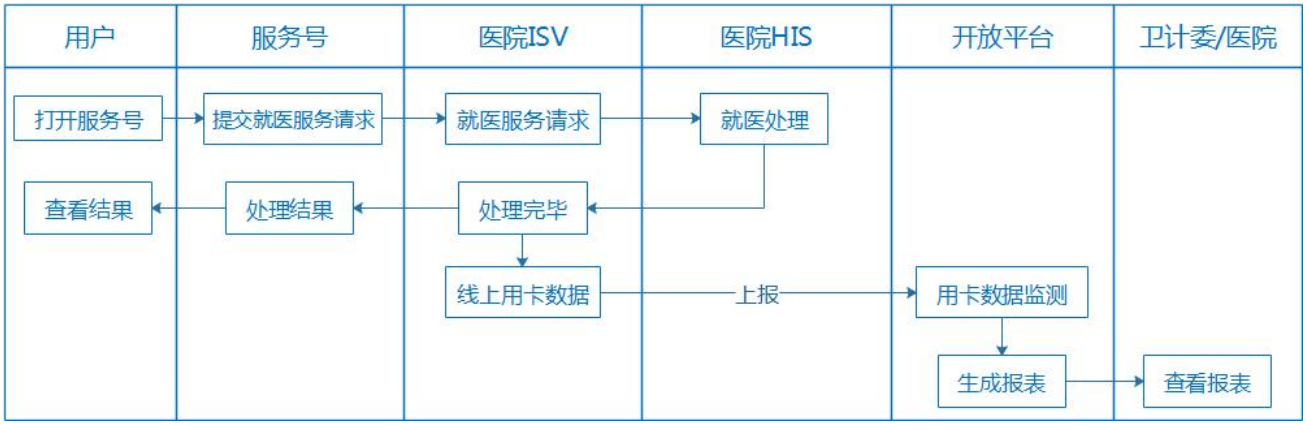

图 2.19 线上用卡数据监测流程图

流程说明

1、用户点击医院服务号就医服务菜单（如挂号、查询报告等），填写并提交就医服务请求，ISV 透传请求给 HIS，HIS 处理请求并返回结果给 ISV，ISV 保存用卡成功的数据；

2、ISV 每日实时或定时调用开放平台 用卡数据监测接口，将用卡数据上报到开放平台；

3、开放平台根据用卡数据每天自动生成统计报表，卫计委/医院在官网登陆即可查看用卡数据报表。

2.7.2 线下用卡数据监测业务流程

线下用卡数据是用户在线下以健康卡为介质，使用各种就医服务的结果数据，HIS 将数据上报给 ISV，ISV 再上报给开放平台，用于监测医院的用卡情况。

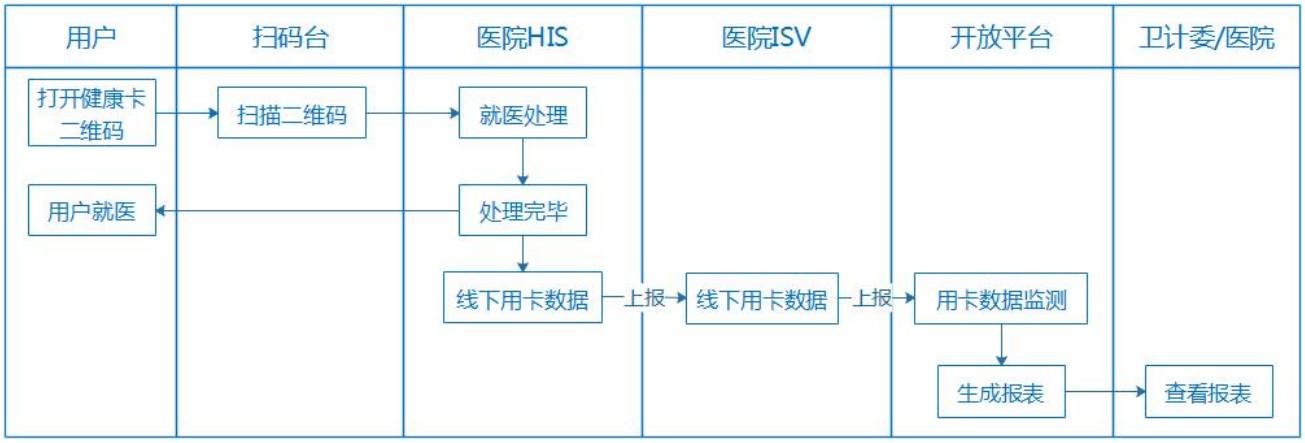

图 2.20 线下用卡数据监测流程图

流程说明

1、用户打开健康卡二维码，扫码台扫描后提交给 HIS 进行处理，HIS 处理请求并返回结果，同时保存用卡成功的数据；

2、HIS 定时将用卡数据上报给 ISV（或者提供接口供 ISV 查询获取），ISV 定时调用开放平台 用卡数据监测接口，将用卡数据上报到开放平台；

3、开放平台根据用卡数据每天自动生成统计报表，卫计委/医院在官网登陆即可查看用卡数据报表。

2.7.3 数据实时传输

系统提供前置接口程序，前置接口程序具备请求转发功能，可调用更高级别的接口完成数据转发，最终上传至国家居民健康卡用卡业务协同与监测系统。

通过实时调用接口(参考接口文档)的方式，可以满足用卡数据实时上传需求。

2.7.4 用卡数据接口

参数名称	参数代码	必选	类型	说明
二维码文本	qrCodeText	条件	string	如果证件类型是健康卡，则二维码数据是必填，反之则二维码数据非必填
身份证号码	idCardNumber	条件	string	如果证件类型不是健康卡，则身份证号码必填，反之则身份证号码非必填
姓名	name	条件	string	如果证件类型不是健康卡，则姓名必填，反之则姓名非必填
时间	time	是	string	格式:yyyy-MM-dd HH:mm:ss
医院 ID	hospitalId	是	string	由医院扫码授权后生成
用卡环节	scene	是	string	参考用卡环节表
用卡科室	department	条件	string	如果用卡环节代码为 010101(挂号)、0101011(预约挂号)、0101012(当日挂号)，则用卡科室数据为必填。参考标准科室表
证件类型	cardType	是	string	01-居民身份证，其他参考证件类型表
用卡渠道	cardChannel	是	string	服务号、app、窗口等

2.7.5 卡使用数据分析

对卡使用情况进行展示，包括但不限于以下指标：

- 全省/各市用卡次数
- 全省/各市用卡趋势
- 已发放卡使用率
- 已发放卡均使用次数
- 健康卡、诊疗卡使用占比
- 持卡就诊人群占比
- 就诊次均刷卡次数统计

对以上指标，均支持多维度、多条件查询，包括所属区域、用卡时间、用卡机构等。

2.7.6 卡使用行为分析

对卡使用行为进行分析展示，包括但不限于以下指标：

- 用卡人群占比统计
- 用卡科室占比统计
- 就诊类型占比统计
- 刷卡操作占比统计
- 受理终端占比统计

对以上指标，均支持多维度、多条件查询，包括所属区域、用卡时间、用卡机构等。

系统提供实时展示功能，可设置指标为实时展示。设置指标实时展示后，将以每秒进行刷新，动态展示指标数据。

系统提供多级下转功能。

若有其它指标需要展示，可通过报表设计器自行生成展示。

2.8 电子健康卡检测流程

开放平台为了保障医院服务号的电子健康卡功能质量，制定了一套流程规范，对上线的服务号进行审核。审核的内容包括：安全检测，接口检测，UI 检测，稳定性检测，性能效率检测。

2.8.1 业务说明

开发商对接开放平台的流程完成后，需要在官网填写流程相关信息提交上线审核，填写内容包括页面体验 URL、体验视频、体验图片。开放平台按照规范进行审核，达标的服务号才能上线对外开放使用。

图 2.21 医院上线审核页面

2.8.2 业务流程

开发商上线审核流程如下：

- 开发商开发完成并在开放平台提交上线审核；
- 提交的页面体验 URL、体验视频、体验图片包括以下功能：新用户建卡、跨院用户一键注册健康卡、老用户批量升级、用卡监测、统一卡包、扫码就医；
- 开放平台审核公众号上线资质，审核功能的内容包括：安全检测，接口检测，UI 检测，稳定性检测，性能效率检测；
- 健康卡开放平台审核不通过，则驳回，开发商优化功能或补充材料后重新提交审核；审核通过，推送给 API 授权管理系统取得授权码。
- 开发商取得授权码后，正式上线运行。

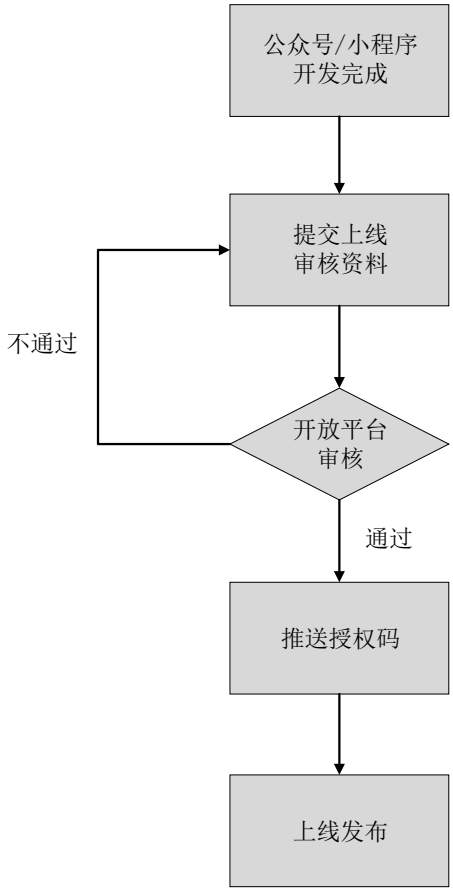

图 2.22 开发商上线审核流程

2.8.3 检测要求

2.8.3.1 协议规范

采用 HTTPS 协议：微信电子健康卡开放平台采用 HTTPS 传输协议。HTTPS 超文本传输安全协议（Hypertext Transfer Protocol Secure，常称为 HTTP over SSL）是一种通过计算机网络进行安全通信的传输协议。HTTPS 经由 HTTP 进行通信。

采用 JSON 数据结构：微信电子健康卡开放平台采用 JSON 数据结构。JSON (JavaScript Object Notation) 是一种轻量级的数据交换格式。它基于 ECMAScript (欧洲计算机协会制定的 JS 规范)的一个子集，采用完全独立于编程语言的文本格式来存储和表示数据。简洁和清晰的层次结构使得 JSON 成为理想的数据交换语言。 易于人阅读和编写，同时也易于机器解析和生成，并有效地提升网络传输效率。

2.8.3.2 功能检测规范

功能检测方面，我们针对医院服务号中关于电子健康卡部分的功能内容进行检测。

编 号	检测项	检测方法及步骤	检测说明
1	电子健康卡二维码申请功能	验证电子健康卡微信服务号是否具备二维码申请功能，申请信息应包含就诊人类型、姓名、性别、证件类型、经验证的证件号码、经验证的手机号等必填项，及医保类型、所在区域、地址等可选项	必选项
2	电子健康卡二维码接收功能	验证电子健康卡微信服务号是否具备二维码接收功能	必选项
3	电子健康卡二维码显示功能	验证电子健康卡微信服务号是否具备二维码显示功能	必选项
4	用户身份认证功能	验证电子健康卡微信服务号是否提供多种用户身份认证功能，如静态口令身份验证功能、动态口令身份验证功能、生物识别身份验证功能、基于密钥身份认证功能等。	必选项
5	就诊卡账户绑定功能	验证电子健康卡微信服务号是否具备就诊卡账户绑定功能（如无此功能，则为不适用）。	条件必选项
6	就诊信息查询	验证电子健康卡微信服务号是否具备就诊信息查询功能（如无此功能，则为不适用）。	条件必选项
7	电子健康卡解绑功能	验证电子健康卡微信服务号是否具备和电子健康卡管理信息系统系统解绑的功能	必选项

2.8.3.3 页面安全检测

编号	检测项	检测方法及步骤	检测说明
1	访问方式	微信服务号通过 HTTPS 方式传输，使用国家认可的加密方式，推荐使用国密算法	必选项
2	页面注入防范	检查页面不存在 SQL 注入、LDAP 注入等漏洞	必选项

2.8.3.4 接入安全检测

编号	检测项	检测方法及步骤	检测说明
1	基本授权	1. 检查电子健康卡微信服务号是否在电子健康卡 SDK 授权管理系统进行注册，并下载 app_secret。 2. 检查电子健康卡微信服务号调用 SDK 时是否符合附录中的 SDK 验证流程。 如移动智能终端应用软件及后台未采用 SDK 方式，则为不适用。	条件必选项

2.8.3.5 二维码安全检测

编号	检测项	检测方法及步骤	检测说明
1	二维码申请	检查移动终端从后台服务器获取条码时，后台是否对用户身份和移动终端进行身份验证。	必选项
2	二维码显示	验证二维码是否动态定时刷新	必选项

2.8.3.6 接口检测规范

编号	检测项	检测方法及步骤	检测说明
1	与电子健康卡管理信息系统接口	电子健康卡微信服务号应正确实现与系统的接口交互，且具备异常处理能力。	必选项
2	接口数据完整性	电子健康卡微信服务号应提供保障接口数据的完整性和异常检查能力。	必选项
3	接口报文功能性	电子健康卡微信服务号应正确实现报文交互和解析功能。	必选项

2.8.3.7 UI 检测规范

编号	检测项	检测方法及步骤	检测说明
1	居民健康卡标识展示	电子健康卡微信服务号应将二维码与居民健康卡标识（LOGO）相结合，展现居民健康卡品牌性。	必选项
2	界面UI布局、功能	电子健康卡微信服务号 UI 界面应在主流手机屏幕上正常展示及实现正确的用户交互操作。	必选项
3	展示完整性	电子健康卡微信服务号应在主流手机屏幕上完整展示信息。	必选项

2.8.3.8 稳定性检测规范

编号	检测项	检测方法及步骤	检测说明
1	操作流畅性	验证电子健康卡微信服务号是否在主流手机上可以实现与用户的流程交互，无严重卡顿、无响应现象。	推荐项
2	健壮性	验证电子健康卡微信服务号是否正确处理各类异常，在大量多次操作下，无强制关闭、异常崩溃、应用无法关闭、弹窗无法关闭等严重问题。	推荐项

2.8.3.9 性能效率检测规范

编 号	检测项	检测方法及步骤	检测说明
1	关键操作耗时	验证电子健康卡微信服务号的生成二维码的整体耗时不超过 3 秒。	必选项

第3章 接口文档

3.1 接口规则

3.1.1 开发指南

- 所有接口调用前，必须要在 **开放配置 - IP 白名单** 中添加服务器公网 IP，否则会报 IP 白名单限制。

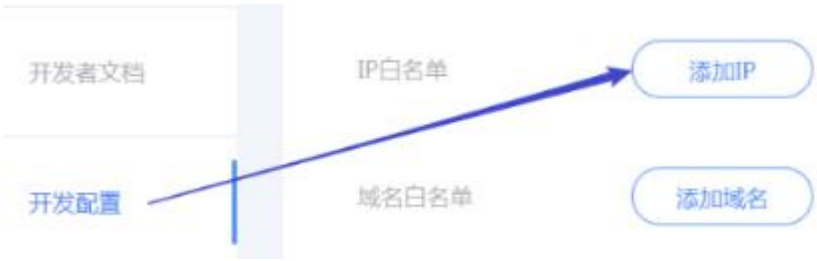

图 3.1 配置 IP 白名单

- 所有授权链接跳转前，，必须要在在官网**开发配置 - 域名白名单**中添回跳页面的域名（支持添加域名+端口&IP），否则会报回跳地址非法。

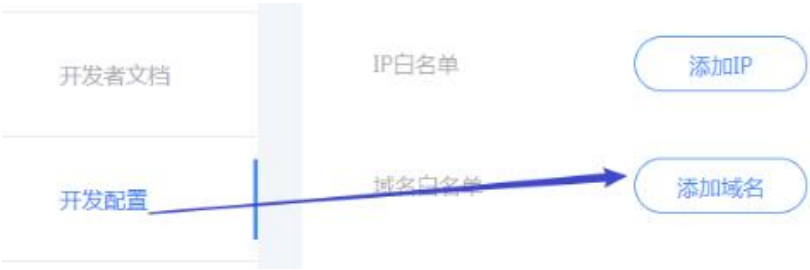

图 3.2 配置域名白名单

- 所有接口输入参数由公共输入参数 commonIn 和请求参数 req 组成，输出参数由公共输出参数 commonOut 和响应参数 rsp 组成。
- 所有接口的请求为 **POST 请求，UTF-8 编码**，请求和响应参数仅支持 **JSON 数据格式**。
- 扫码入驻七天后仍未上线的医院，**限制每个接口调用次数**，具体为 100 次/天/接口，上线后自动解除限制。

3.1.1.1 基本术语

- **HTTPS**：超文本传输安全协议（Hypertext Transfer Protocol Secure，常称为 HTTP over SSL）是一种通过计算机网络进行安全通信的传输协议。HTTPS 经由 HTTP 进行通信，但利用 SSL/TLS 来加密数据包；

- **JSON**：JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式；
- **appId**：由健康卡开放平台分配给各个服务号开发商的用户凭证；
- **appSecret**：由健康卡开放平台分配给各个服务号开发商的用户凭证密钥；
- **appToken**：通过 appId 和 appSecret 向健康卡开放平台换取接口调用凭证；
- **wechatCode**：微信身份码 wechatCode 唯一标识一个微信帐号，用于在开放平台建卡；
- **healthCode**：健康卡授权码 healthCode 对应一张健康卡，用户获取对应健康卡的信息。

3.1.2 公共参数

所有接口的输入参数由公共输入参数 commonIn 和请求参数 req 组成，输出参数由公共输出参数 commonOut 和响应参数 rsp 组成。

公共输入参数 commonIn 说明：

参数名称	参数代码	必选	类型	说明
接口调用凭证	appToken	是	string	详见获取接口调用凭证 appToken 接口
请求 ID	requestId	是	string	防止**重放攻击**和**排查问题**，建议按照 UUID 规则生成，确保唯一性
医院 ID	hospitalId	是	string	由医院扫码授权后生成，操作指引详见【邀请医院入驻】如何邀请医院入驻，授权绑定？
时间戳	timestamp	是	string	Unix 时间戳，精确到秒如 1525392000
请求签名	sign	是	string	详见签名规则

输入参数示例 ：

```
{
    "commonIn": {
        "appToken": "11fd722a113969bf2480fe4781fc7234",
        "requestId": "A37FA9D0D0DF432B9D367B16AEEDE77A",
        "hospitalId": "10086",
        "timestamp": "1525392000",
        "sign": "Q5vp1tdaHjuQpDK8yuDOAzFKTOQs5PxgzhLbxMpnadE="
    },
    "req": {
        ......//请求参数
    }
}
```

公共输出参数 commonOut 说明：

参数名称	参数代码	必选	类型	说明
请求 ID	requestId	是	string	对应 commonIn 中的 requestId
状态码	resultCode	是	int	详见全局返回码
状态码描述	errMsg	是	string	详见全局返回码

输出参数示例：

```
{
    "commonOut": {
        "requestId": "A37FA9D0D0DF432B9D367B16AEEDE77A",
        "resultCode": 0,
        "errMsg": "成功"
    },
    "rsp": {
        ......//响应参数
    }
}
```

3.1.3 签名规则

服务号开发商调用接口时的请求签名由公共参数+接口输入参数+appSecret 生成，假定 appSecret：8c8e763f443ef983ac33aef1c7085cfb。

以 获取接口调用凭证 appToken 为例，当调用这一接口时请求参数包括：requestId、hospitalId、timestamp、appId（appToken 为空串，不参与签名，其他空值的参数同样不参与签名），而 sign 参数 正是由这些参数共同生成。

3.1.3.1 对参数排序

首先需要对所有请求参数按参数名做 **字典序升序** 排列，上述示例参数的排序结果为：

注意：**注册批量健康卡** registerBatchHealthCard 接口中，请求参数含有嵌套的 JSON 字段，也需要按参数名做进行 **字典序升序** 排列。

所谓字典序升序排列，直观上就如同在字典中排列单词一样排序，按照字母表或数字表里递增顺序的排列次序，即先考虑第一个“字母”，在相同的情况下考虑第二个“字母”，依此类推。

```
 {
     "appid":"a1a2e0bde41574ad8ea9a4bb58022oop",
     "hospitalId":"90003",
     "requestId":"DB4D975748A84309977EA25224C0F5CF",
```

```
    "timestamp":"1525392000",
}
```

3.1.3.2 拼接签名原文

将上一步排序好的请求参数格式转化成 **参数名称=参数值** 的形式，如其参数名称为 appId，参数值为 a1a2e0bde41574ad8ea9a4bb58022oop，因此格式化后就为 appId=a1a2e0bde41574ad8ea9a4bb58022oop。

将上述示例参数拼接后的签名原文为：

appId=a1a2e0bde41574ad8ea9a4bb58022oop&hospitalId=90003&requestId=DB4D975748A84309977EA25224C0F5CF&timestamp=1525392000

3.1.3.3 生成签名串

将上一步中获得的签名原文与 appSecret 进行字符串拼接，即：签名原文+appSecret，然后使用 SHA256 算法 对其加密，将生成的签名串（字节数组）使用 Base64 编码，即可获得最终的签名串。

伪代码如下：

```
签名原文 = "appId=a1a2e0bde41574ad8ea9a4bb58022oop&hospitalId=90003&requestId=DB4D975748A84309977EA25224C0F5CF&timestamp=1525392000"
签名 = Base64(SHA256("appId=a1a2e0bde41574ad8ea9a4bb58022oop&hospitalId=90003&requestId=DB4D975748A84309977EA25224C0F5CF&timestamp=15253920008c8e763f443ef983ac33aef1c7085cfb"))
```

最终得到的签名串为：

TsccMUMTHfOiEovR2hMlRXcQqctRFmPbpPIZdxXCJ/o=

3.1.3.4 Java 签名 DEMO

代码示例：

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.alibaba.fastjson.serializer.SerializerFeature;
import java.math.BigDecimal;
import java.math.BigInteger;
import java.security.MessageDigest;
import java.util.*;

public class SignDemo {

    public static void main(String[] args) {
```

```
    //appSecret
    String appSecret = "8c8e763f443ef983ac33aef1c7085cfb";

    // 示例：获取接口调用凭证 appToken 接口的完整请求参数如下
    String reqParams = "{" +
            "    \"commonIn\":{" +
            "        \"appToken\":\"\"," +
            "        \"requestId\":\"DB4D975748A84309977EA25224C0F5CF\"," +
            "        \"hospitalId\":\"90003\"," +
            "        \"timestamp\":\"1525392000\"," +
            "        \"sign\":\"\"" +
            "    }," +
            "    \"req\":{" +
            "        \"appId\":\"a1a2e0bde41574ad8ea9a4bb58022oop\"" +
            "    }" +
            "}";

     //构造当前时间戳
    long time = System.currentTimeMillis();
    String nowTimeStamp = String.valueOf(time / 1000);

    //构造 requestId
    String requestId = UUID.randomUUID().toString().replaceAll("-", "");

    JSONObject jsonObject = JSON.parseObject(reqParams);

    Map<String, Object> commonIn = jsonObject.getJSONObject("commonIn");
    //commonIn.put("timestamp", nowTimeStamp);
    //commonIn.put("requestId", requestId.toUpperCase());
    Map<String, Object> req = jsonObject.getJSONObject("req");

    // 生成原始签名串
    SortedMap<String, Object> treeMap = new TreeMap<>();
    treeMap.putAll(commonIn);
    treeMap.putAll(req);
    String rawStr=getParamsFromMap(treeMap);

    // 生成签名
    String sign = generateSign(rawStr, appSecret);
    System.out.println(sign);
}

/**
 * 对请求参数进行排序拼接(含嵌套的 JSON 字符串)，生成待签名字符串
```

```
     *
     * @param map
     * @return
     */
    private static String getParamsFromMap(SortedMap<String, Object> map) {
        // sign 不参与签名
        map.remove("sign");
        StringBuilder sb = new StringBuilder();
        Set es = map.entrySet();
        Iterator it = es.iterator();
        while (it.hasNext()) {
            Map.Entry entry = (Map.Entry) it.next();
            String k = entry.getKey().toString();
            Object objVal = entry.getValue();
            if (objVal == null) { //值为空的参数不参与签名
                continue;
            }
            String v;
            if (isBaseDataType(objVal.getClass())) {
                v = objVal.toString();
            } else {
                v = JSON.toJSONString(objVal, SerializerFeature.MapSortField);
            }
            if (!v.equals("")) {
                if (it.hasNext()) {
                    sb.append(k).append("=").append(v).append("&");
                } else {
                    sb.append(k).append("=").append(v);
                }
            }
        }
        return sb.toString();
    }

    private static boolean isBaseDataType(Class clazz) {
        return (clazz.equals(String.class) || clazz.equals(Integer.class) || clazz.equal
s(Byte.class)
                || clazz.equals(Long.class) || clazz.equals(Double.class) || clazz.equal
s(Float.class)
                || clazz.equals(Character.class) || clazz.equals(Short.class) || clazz.e
quals(BigDecimal.class)
                || clazz.equals(BigInteger.class) || clazz.equals(Boolean.class) || claz
z.equals(Date.class) || clazz
                .isPrimitive());
```

```
    }

    /**
     * Base64(sha256(rawStr + appSecret))进行签名
     * @param rawStr      原始字符串
     * @param appSecret   密钥
     */
    private static String generateSign(String rawStr, String appSecret) {
        try {
            // 先用sha256加密
            MessageDigest sha256 = MessageDigest.getInstance("SHA-256");
            sha256.update((rawStr + appSecret).getBytes("utf-8"));

            // 再用base64编码
            return Base64.getEncoder().encodeToString(sha256.digest());
        } catch (Exception e) {
            e.printStackTrace();
        }
        return null;
    }
}
```

3.1.3.5 C#签名 DEMO

代码示例

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Text;
using Newtonsoft.Json;
using Newtonsoft.Json.Serialization;
using System.Security;
using System.Security.Cryptography;
using System.Linq;

namespace Demo
{
    class Program
    {
        static void Main(string[] args)
        {
            string appId = "a1a2e0bde41574ad8ea9a4bb58022oop";
            string secret = "8c8e763f443ef983ac33aef1c7085cfb";
            string appToken = "c37439bbf711b023af8fed1ab1f06f43";
```

```
            //示例：注册健康卡接口的完整请求参数如下
            IDictionary<string, Object> commonIn = new Dictionary<string, Object>();
            commonIn.Add("hospitalId", "20007");
            commonIn.Add("appToken", appToken);
            commonIn.Add("requestId", "2a9a8ff876e7f3e1df1a2490a29b22a1");
            commonIn.Add("timestamp", "1536810104");

            IDictionary<string, Object> req = new Dictionary<string, Object>();

            /*  复杂参数
            SortedDictionary<string,Object> cardInfo = new SortedDictionary<string,Objec
t>();
            cardInfo.Add("name","张三");
            cardInfo.Add("idCard","430203188808084321");
            cardInfo.Add("gender","男");
            cardInfo.Add("nation","汉族");
            cardInfo.Add("birthday", "1999-03-23");
            cardInfo.Add("address", "腾讯大厦");
            req.Add("cardInfo", cardInfo);*/

            req.Add("appId", appId);

            string sign = getSign(commonIn, req, secret);
            commonIn.Add("sign", sign);

            Console.WriteLine("sign:" + sign);

            IDictionary<string, Object> paramDic = new Dictionary<string, Object>();
            paramDic.Add("commonIn", commonIn);
            paramDic.Add("req", req);

            string jsonParam = JsonConvert.SerializeObject(paramDic);

            Console.WriteLine("json param:" + jsonParam);

        }

        private static string getSign(IDictionary<String, Object> commonIn, IDictionary<
String, Object> req, string secret)
        {
            SortedDictionary<String, Object> sortParamDic = new SortedDictionary<String,
 Object>(commonIn);
            foreach (KeyValuePair<string, Object> kv in req)
```

```
        {
            sortParamDic.Add(kv.Key, kv.Value);
        }

        string paramStr = getParamStr(sortParamDic);

        Console.WriteLine("param str:" + paramStr);

        byte[] bytes = Encoding.UTF8.GetBytes(paramStr + secret);
        byte[] hash = SHA256Managed.Create().ComputeHash(bytes);

        string sign = Convert.ToBase64String(hash);
        return sign;
    }

    private static string getParamStr(SortedDictionary<String, Object> sortParamDic)
    {
        StringBuilder stringBuilder = new StringBuilder();
        foreach (KeyValuePair<string, Object> kv in sortParamDic)
        {
            if (kv.Value == null)
            {
                continue;
            }
            string val;
            if (isBaseType(kv.Value))
            {
                val = kv.Value.ToString();
            }
            else
            {
                val = JsonConvert.SerializeObject(kv.Value);
                Console.WriteLine(val);
            }

            if (val.Trim().Equals(""))
            {
                continue;
            }

            if (stringBuilder.Length > 0)
            {
                stringBuilder.Append("&");
            }
```

```
                stringBuilder.Append(kv.Key);
                stringBuilder.Append("=");
                stringBuilder.Append(val);
            }

            return stringBuilder.ToString();
        }

        private static bool isBaseType(Object obj)
        {
            Type type = obj.GetType();
            return type.IsPrimitive || type.Equals(typeof(string));
        }
    }
}
```

3.1.4 SDK

使用 SDK （点击下载 open-platform-sdk.jar） 测试接口，不需要构造签名、时间戳，直接调用接口方法即可测试。调用接口方法后会输出日志到控制台，日志包括签名原文，签名密文，输入参数以及输出参数。建议导入该 SDK 进行开发，提高开发速度。示例：获取接口调用凭证 appToken 接口输出的日志截图如下：

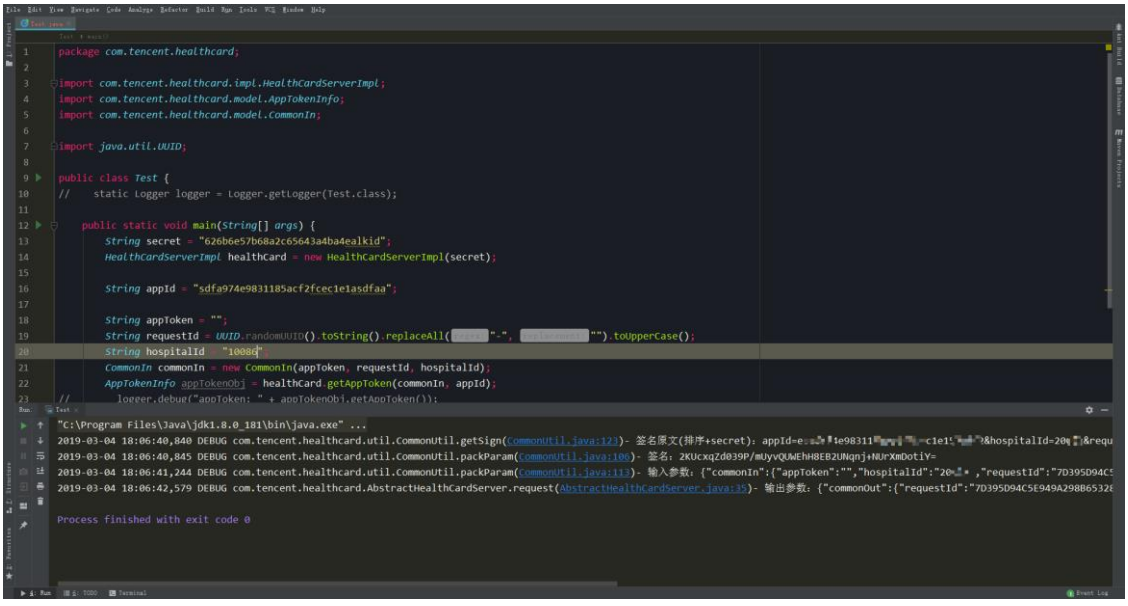

图 4.3 接口测试 SDK 示例图

3.1.4.1 测试步骤

1. 导入 SDK，其中 com/tencent/healthcard/impl/HealthCardServerImpl.class 中列出了所有的接口方法，导入步骤略；

2. 配置 pom.xml 文件，以 log4j 日志组件为例，需增加下述配置代码：

```
<dependency>
<groupId>org.slf4j</groupId>
<artifactId>slf4j-api</artifactId>
<version>1.7.25</version>
</dependency>
<dependency>
<groupId>log4j</groupId>
<artifactId>log4j</artifactId>
<version>1.2.17</version>
</dependency>
<dependency>
<groupId>org.slf4j</groupId>
<artifactId>slf4j-log4j12</artifactId>
<version>1.7.21</version>
</dependency>
```

3. 新建并配置 log4j.properties 文件，文件相对路径为...\demo\src\main\resources\log4j.properties，配置示例如下：

```
## 设置###
log4j.rootLogger = debug,stdout
log4j.logger.com.tencent.healthcard=debug

### 输出信息到控制抬 ###
log4j.appender.stdout = org.apache.log4j.ConsoleAppender
log4j.appender.stdout.Target = System.out
log4j.appender.stdout.layout = org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern = %d %-5p %l- %m%n
```

4. 调用 获取接口调用凭证 appToken 接口 demo，代码中的 appId、appSecret、hospitalId 为虚构，请使用开放平台分配的 appId、appSecret、hospitalId，示例代码如下：

```
package com.tencent.healthcard;

import com.tencent.healthcard.impl.HealthCardServerImpl;
import com.tencent.healthcard.model.AppTokenInfo;
import com.tencent.healthcard.model.CommonIn;
```

```
import java.util.UUID;

public class Test {
//    static Logger logger = Logger.getLogger(Test.class);

    public static void main(String[] args) {
        //appSecret
        String secret = "626b6e57b68a2c65643a4ba4ealkid";
        //创建健康卡实例，传入 appSecret
        HealthCardServerImpl healthCard = new HealthCardServerImpl(secret);

        //构造【获取接口调用凭证 appToken 接口】公共输入参数 commonIn 参数
        String appToken = "";
        String requestId = UUID.randomUUID().toString().replaceAll("-", "").toUpperCase
();
        String hospitalId = "10086";

        //创建【公共输入参数 commonIn】实例
        CommonIn commonIn = new CommonIn(appToken, requestId, hospitalId);

        //构造【获取接口调用凭证 appToken 接口】请求参数 req
        String appId = "sdfa974e9831185acf2fcec1e1asdfaa";

        //调用接口方法
        AppTokenInfo appTokenObj = healthCard.getAppToken(commonIn, appId);
//        logger.debug("appToken: " + appTokenObj.getAppToken());
    }
}
```

5.在控制台可以看到输出的日志，示例如下：

```
2019-03-04 18:06:40,840 DEBUG com.tencent.healthcard.util.CommonUtil.getSign(CommonUtil.
java:123)- 签名原文(排序+secret)：appId=sdfa974e9831185acf2fcec1e1asdfaa&hospitalId=1008
6&requestId=7D395D94C5E949A298B653285F9133E3&timestamp=15516940000626b6e57b68a2c65643a4b
a4ealkid
2019-03-04 18:06:40,845 DEBUG com.tencent.healthcard.util.CommonUtil.packParam(CommonUti
l.java:106)- 签名：2KUcxqZdO39P/mUyvQUWEhH8EB2UNqnj+NUrXmDotiY=
2019-03-04 18:06:41,244 DEBUG com.tencent.healthcard.util.CommonUtil.packParam(CommonUti
l.java:113)- 输入参数：{"commonIn":{"appToken":"","hospitalId":"10086","requestId":"7D39
5D94C5E949A298B653285F9133E3","sign":"2KUcxqZdO39P/mUyvQUWEhH8EB2UNqnj+NUrXmDotiY=","tim
estamp":"1551694000"},"req":{"appId":"sdfa974e9831185acf2fcec1e1asdfaa"}}
2019-03-04 18:06:42,579 DEBUG com.tencent.healthcard.AbstractHealthCardServer.request(Ab
stractHealthCardServer.java:35)- 输出参数：{"commonOut":{"requestId":"7D395D94C5E949A298
B653285F9133E3","errMsg":"success","resultCode":0},"rsp":{"expiresIn":7200,"appToken":"4
fa6b9453a8e486dc6c1ba58ecf46fd2"}}
```

3.2 接口列表

3.2.1 获取接口调用凭证 appToken 接口

接口地址：https://p-healthopen.tengmed.com/rest/auth/HealthCard/HealthOpenAuth/AuthObj/getAppToken

该接口用于获取 appToken，appToken 是其他所有接口的调用凭证，是 commonIn 参数之一，通过该接口获取 appToken 时，appToken 参数不校验，可以为空；appToken 有效期 7200 秒，有效期内多次调用该接口，会导致 token 刷新，旧 token 将失效。

请求参数 req 说明：

参数名称	参数代码	必选	类型	说明
服务号开发商用户凭证	appId	是	string	开放平台官网分配的 appId

输入参数示例：

```
{
    "commonIn":{
        "appToken":"",
        "requestId":"E856812660584E208D7421E1CAACE8C3",
        "hospitalId":"00000",
        "timestamp":"1525392000",
        "sign":"Q5vp1tdaHjuQpDK8yuDOAzFKTOQs5PxgzhLbxMpnadE="
    },
    "req":{
        "appId":"d7656ef9ab5eb27c01724cd37079d709"
    }
}
```

响应参数 rsp 说明：

参数名称	参数代码	必选	类型	说明
接口调用凭证	appToken	是	string	其他所有接口的调用凭证
凭证有效时间	expiresIn	是	int	appToken 有效时间，默认为 7200 秒

输出参数示例：

```
{
    "commonOut":{
        "requestId":"E856812660584E208D7421E1CAACE8C3",
        "resultCode":0,
        "errMsg":"成功"
```

```
    },
    "rsp":{
        "appToken":"9A60D3D561E4ADBA89ACD9FBF2FB8D2C",
        "expiresIn":7200
    }
}
```

SDK 中调用接口示例：

```
import com.tencent.healthcard.impl.HealthCardServerImpl;
import com.tencent.healthcard.model.AppTokenInfo;
import com.tencent.healthcard.model.CommonIn;
import java.util.UUID;

public class GetAppToken {
    public static void  main(String[] args) {
        //appSecret
        String appSecret="0626b6e57b68a2c65643a4ba4e084884";
        HealthCardServerImpl healthCard=new HealthCardServerImpl(appSecret);

        //示例 1：调用获取接口调用凭证 appToken 接口
        //构造公共输入参数 commonIn
        String appToken="";
        String requestId= UUID.randomUUID().toString().replaceAll("-", "").toUpperCase
();
        String hospitalId="20012";
        CommonIn commonIn=new CommonIn(appToken,requestId,hospitalId);
        //构造请求参数 req
        String appId="e8e8974e9831185acf2fcec1e157cd52";
        //调用接口
        AppTokenInfo appTokenInfo=healthCard.getAppToken(commonIn,appId);
        //打印响应
        System.out.println(appTokenInfo.getAppToken());
    }
}
```

3.2.2 注册健康卡接口

接口地址：`https://p-healthopen.tengmed.com/rest/auth/HealthCard/HealthOpenPlatform/ISVOpenObj/registerHealthCard`

该接口用于注册健康卡，注册时会对用户身份进行实名核身，适用于 新用户建卡流程，请求参数为用户身份信息，响应参数为二维码文本数据、健康卡 ID、健康卡主索引、扩展字段。

请求参数 req 说明：

参数名称	参数代码	必选	类型	说明
微信身份码	wechatCode	是	string	获取方式见 【新用户建卡流程】获取用户微信身份码 wechatCode
姓名	name	是	string	
性别	gender	是	string	男、女
民族	nation	是	string	汉族、满族等
出生年月日	birthday	是	string	格式：yyyy-MM-dd
证件号码	idNumber	是	string	
证件类型	idType	是	string	01-居民身份证，其他参考证件类型表
地址	address	否	string	
联系方式1	phone1	是	string	
联系方式2	phone2	否	string	
院内 ID	patid	否	string	院内用户唯一标识

输入参数示例：


```
{
    "commonIn": {
        "appToken": "c86aae0f838f2018a6f6246d0bbcecd5",
        "requestId": "4D6FFFE544AE4CE1B5E5FA2DC1566E1C",
        "hospitalId": "00000",
        "timestamp": "1525392000",
        "sign": "Q5vp1tdaHjuQpDK8yuDOAzFKTOQs5PxgzhLbxMpnadE="
    },
    "req": {
        "wechatCode": "微信身份授权码",
        "name": "张三",
        "gender": "男",
        "nation": "汉族",
        "birthday": "1998-09-08",
        "idNumber": "身份证号码",
        "idType": "01",
        "address": "地址",
        "phone1": "18808808808",
        "phone2": "18808808808",
        "patid": "1003243"
    }
}
```

响应参数 rsp 说明：

参数名称	参数代码	必选	类型	说明
二维码文本	qrCodeText	是	string	根据当地卫计委要求返回静态码或动态码，一般返回静态码，静态码由【健康卡 ID+“:1”】组成
健康卡 ID	healthCardId	否	string	当二维码文本返回静态码时，健康卡 ID 不返回值，此时可截取静态码文本前 64 位字符串获取健康卡 ID
健康卡主索引	phid	是	string	
扩展字段	adminExt	是	string	返回卡管平台注册接口响应包的全部内容（不含图片数据）

输出参数示例：


```
{
    "commonOut":
    {
        "requestId": "4D6FFFE544AE4CE1B5E5FA2DC1566E1C",
        "resultCode": 0,
        "errMsg": "成功"
    },
    "rsp":
    {
        "qrCodeText": "C220AE414CE6EE581037C311AE24518FCFE19C429BECD478C1A13976260FXXXX:
1",
        "healthCardId": "C220AE414CE6EE581037C311AE24518FCFE19C429BECD478C1A13976260FXXX
X",
        "phid": "584AF4B1D110B8BB7CBAC978C03B657191DF84863144E436B5CA38C0F0E2XXXX",
        "adminExt":"{\"qr_code\":\"C220AE414CE6EE581037C311AE24518FCFE19C429BECD478C1A13
976260FXXXX:1\",\"ecardId\":\"C220AE414CE6EE581037C311AE24518FCFE19C429BECD478C1A1397626
0FXXXX\",\"main_index\":\"584AF4B1D110B8BB7CBAC978C03B657191DF84863144E436B5CA38C0F0E2XX
XX\",\"id_type\":\"居民身份证\",\"id_number\":\"身份证号码\",\"name\":\"姓名\",\"sex\":\
"性别\",\"birthday\":\"1998-09-08\",\"telephone\":\"手机号码\",\"nation\":\"民族\",\"uni
t\":\"null\",\"address\":\"地址\"}"
    }
}
```


SDK 中调用接口示例

```
import com.tencent.healthcard.impl.HealthCardServerImpl;
import com.tencent.healthcard.model.CommonIn;
import com.tencent.healthcard.model.HealthCardInfo;
import java.util.UUID;
```

```
public class RegisterHealthCard {
    public static void  main(String[] args) {
        //appSecret
        String appSecret="0626b6e57b68a2c65643a4ba4e0xxxxx";
        HealthCardServerImpl healthCard=new HealthCardServerImpl(appSecret);
        //示例 2：注册健康卡接口
        //构造公共输入参数 commonIn
        String appToken="5686d520fe95578a93b618282fexxxxx";
        String requestId= UUID.randomUUID().toString().replaceAll("-", "").toUpperCase
();
        String hospitalId="10086";
        CommonIn commonIn=new CommonIn(appToken,requestId,hospitalId);
        //构造请求参数 req
        String birthday="1996-01-23";
        String idType="01";
        String address="广东省深圳市腾讯大厦";
        String gender="男";
        String nation="汉族";
        String name="小邓";
        String wechatCode="73EFA6796D3869FF82FAE7E81E9XXXX";
        String idNumber="432624198888883116";
        String phone1="18565818888";
        String phone2="18565818899";
        String patid="1003243";
        HealthCardInfo healthCardInfoReq=new HealthCardInfo();
        healthCardInfoReq.setAddress(address);
        healthCardInfoReq.setBirthday(birthday);
        healthCardInfoReq.setGender(gender);
        healthCardInfoReq.setIdNumber(idNumber);
        healthCardInfoReq.setIdType(idType);
        healthCardInfoReq.setNation(nation);
        healthCardInfoReq.setName(name);
        healthCardInfoReq.setPhone1(phone1);
        healthCardInfoReq.setPhone2(phone2);
        healthCardInfoReq.setWechatCode(wechatCode);
        healthCardInfoReq.setWechatCode(patid);
        //调用接口
        HealthCardInfo healthCardInfoRsp=healthCard.registerHealthCard(commonIn,healthCa
rdInfoReq);
        //打印响应
        System.out.println(healthCardInfoRsp.getQrCodeText());
        System.out.println(healthCardInfoRsp.getPhid());
        System.out.println(healthCardInfoRsp.getHealthCardId());
        System.out.println(healthCardInfoRsp.getAdminExt());
```

```
    }
}
```

3.2.3 通过健康卡授权码获取健康卡接口

接口地址：`https://p-healthopen.tengmed.com/rest/auth/HealthCard/HealthOpenPlatform/ISVOpenObj/getHealthCardByHealthCode`

该接口用于获取用户健康卡信息，适用于 跨院用户一键注册健康卡 ，请求参数为健康卡授权码 healthCode，响应参数为用户信息。

请求参数 req 说明：

参数名称	参数代码	必选	类型	说明
健康卡授权码	healthCode	是	string	获取方式见 【跨院用户一键注册健康卡】获取用户健康卡授权码 healthCode

输入参数示例：

```
{
    "commonIn": {
        "appToken": "c86aae0f838f2018a6f6246d0bbcecd5",
        "requestId": "4D6FFFE544AE4CE1B5E5FA2DC1566E1C",
        "hospitalId": "00000",
        "timestamp": "1525392000",
        "sign": "Q5vp1tdaHjuQpDK8yuDOAzFKTOQs5PxgzhLbxMpnadE="
    },
    "req": {
        "healthCode": "F9D3F8A308FC0EABC581F5903CAA1094"
    }
}
```

响应参数 rsp 说明：

参数名称	参数代码	必选	类型	说明
健康卡数据	card	是	obj	json 结构体，内容如下
二维码文本	qrCodeText	是	string	动态二维码或静态二维码
姓名	name	是	string	
性别	gender	是	string	男、女
民族	nation	是	string	汉族、满族等

出生年月日	birthday	否	string	格式：yyyy-MM-dd
证件号码	idNumber	是	string	
证件类型	idType	是	string	01-居民身份证，其他参考证件类型表
地址	address	否	string	
联系方式 1	phone1	是	string	
联系方式 2	phone2	否	string	
主索引	phid	是	string	
院内 ID	patid	否	string	院内用户唯一标识，根据 commonIn 的 hospitalId 返回
健康卡 ID	healthCardId	否	string	如果 qrCodeText 是动态二维码，则 healthCardId 有值
扩展字段	adminExt	是	string	返回卡管平台查询接口响应包的全部内容

输出参数示例：


```
{
    "commonOut": {
        "requestId": "4D6FFFE544AE4CE1B5E5FA2DC1566E1C",
        "resultCode": 0,
        "errMsg": "成功"
    },
    "rsp": {
        "card": {
            "qrCodeText": "C7DA29345B6DF90A6F5BBEBD73EBE2EDA26F341A6CFEEEB121XXX:1",
            "name": "张三",
            "gender": "男",
            "nation": "汉族",
            "birthday": "1998-09-08",
            "idNumber": "101102199809089988",
            "idType": "01",
            "address": "地址",
            "phone1": "18808808808",
            "phone2": "18808808808",
            "phid": "360782201808181131100003XXXX",
            "patid": "院内 ID",
            "healthCardId":"A4E0374BCBEAB40E4D66AB5078B44XXX",
            "adminExt":"{\"qr_code\":\"C7DA29345B6DF90A6F5BBEBD73EBE2EDA26F341A6CFEEEB121XXX:1\",\"ecardId\":\"C7DA29345B6DF90A6F5BBEBD73EBE2EDA26F341A6CFEEEB121XXX\",\"main_index\":\"360782201808181131100003XXXX\",\"id_type\":\"居民身份证\",\"id_number\":\"101102199809089988\",\"name\":\"张三\",\"sex\":\"男\",\"birthday\":\"1998-09-08\",\"telephone\":\"18808808808\",\"nation\":\"汉族\",\"unit\":\"null\",\"address\":\"地址\"}"
        }
    }
}
```

3.2.4 通过健康卡二维码获取健康卡接口

接口地址： https://p-healthopen.tengmed.com/rest/auth/HealthCard/HealthOpenPlatform/ISVOpenObj/getHealthCardByQRCode

该接口用于获取用户健康卡信息，适用于 【扫码就医流程(可选)】1、静态二维码就医流程 ，请求参数为二维码文本数据，响应参数为用户信息。

请求参数 req 说明：

参数名称	参数代码	必选	类型	说明
二维码文本	qrCodeText	是	string	动态二维码或静态二维码

输入参数示例：

```
{
    "commonIn": {
        "appToken": "c86aae0f838f2018a6f6246d0bbcecd5",
        "requestId": "4D6FFFE544AE4CE1B5E5FA2DC1566E1C",
        "hospitalId": "00000",
        "timestamp": "1525392000",
        "sign": "Q5vp1tdaHjuQpDK8yuDOAzFKTOQs5PxgzhLbxMpnadE="
    },
    "req": {
        "qrCodeText": "C7DA29345B6DF90A6F5BBEBD73EBE2EDA26F341A6CFEEEB121XXX:1"
    }
}
```

响应参数 rsp 说明：

参数名称	参数代码	必选	类型	说明
健康卡数据	card	是	obj	json 结构体，内容如下
二维码数据	qrCodeText	是	string	动态二维码或静态二维码
姓名	name	是	string	
性别	gender	是	string	男、女
民族	nation	是	string	汉族、满族等
出生年月日	birthday	否	string	格式：yyyy-MM-dd
证件号码	idNumber	是	string	

证件类型	idType	是	string	01-居民身份证，其他参考证件类型表
地址	address	否	string	
联系方式 1	phone1	是	string	
联系方式 2	phone2	否	string	
健康卡主索引	phid	是	string	
院内 ID	patid	否	string	院内用户唯一标识，根据 commonIn 的 hospitalId 返回
健康卡 ID	healthCardId	条件	string	如果 qrCodeText 是动态，则 healthCardId 有值
扩展字段	adminExt	是	string	返回卡管平台查询接口响应包的全部内容

输出参数示例：


```
{
    "commonOut": {
        "requestId": "4D6FFFE544AE4CE1B5E5FA2DC1566E1C",
        "resultCode": 0,
        "errMsg": "成功"
    },
    "rsp": {
        "card": {
            "qrCodeText": "C7DA29345B6DF90A6F5BBEBD73EBE2EDA26F341A6CFEEEB121XXX:1",
            "name": "张三",
            "gender": "男",
            "nation": "汉族",
            "birthday": "1998-09-08",
            "idNumber": "101102199809089988",
            "idType": "01",
            "address": "地址",
            "phone1": "18808808808",
            "phone2": "18808808808",
            "phid": "360782201808181131100003XXXX",
            "patid": "院内 ID",
            "healthCardId":"A4E0374BCBEAB40E4D66AB5078B44XXX",
            "adminExt":"{\"qr_code\":\"C7DA29345B6DF90A6F5BBEBD73EBE2EDA26F341A6CFEEEB121XXX:1\",\"ecardId\":\"C7DA29345B6DF90A6F5BBEBD73EBE2EDA26F341A6CFEEEB121XXX\",\"main_index\":\"360782201808181131100003XXXX\",\"id_type\":\"居民身份证\",\"id_number\":\"101102199809089988\",\"name\":\"张三\",\"sex\":\"男\",\"birthday\":\"1998-09-08\",\"telephone\":\"18808808808\",\"nation\":\"汉族\",\"unit\":\"null\",\"address\":\"地址\"}"
        }
    }
}
```


3.2.5 OCR 接口

接口地址：https://p-healthopen.tengmed.com/rest/auth/HealthCard/HealthOpenPlatform/ISVOpenObj/ocrInfo

该接口用于将身份证图像数据识别为文本数据，适用于 新用户建卡流程 ，请求参数为身份证正面照片的 base64 编码数据，响应参数为身份证上的文本信息。

请求参数 req 说明：

参数名称	参数代码	必选	类型	说明
身份证图片	imageContent	是	string	身份证正面照片的 base64 编码数据，头部信息需要删除，如 image/png;base64、image/jpeg/png;base64 等；数据量要压缩到百 K 级别上传

输入参数示例：

```
{
    "commonIn": {
        "appToken": "123",
        "requestId": "123",
        "hospitalId": "00000",
        "timestamp": "1525392000",
        "sign": "Q5vp1tdaHjuQpDK8yuDOAzFKTOQs5PxgzhLbxMpnadE="
    },
    "req": {
        "imageContent": "xxxx"
    }
}
```

响应参数 rsp 说明：

参数名称	参数代码	必选	类型	说明
身份证信息	cardInfo	是	obj	json 结构体，内容如下
姓名	name	是	string	
身份证号	id	是	string	
性别	gender	是	string	男、女
民族	nation	是	string	汉、满等
出生日期	birth	是	string	格式： 1991/5/25

地址	address	是	string	
签发机关	authority	否	string	xxx 公安局
有效期	validDate	否	string	xxxx.xx.xx-xxxx.xx.xx

输出参数示例：


```
{
    "commonOut": {
        "requestId": "123",
        "resultCode": 0,
        "errMsg": "成功"
    },
    "rsp": {
        "cardInfo": {
            "name":"张三",
            "id":"998",
            "gender":"女",
            "nation":"汉",
            "birth":"1991/5/25",
            "address":"xxx",
            "authority":"",
            "validDate":""
        }
    }
}
```


3.2.6 绑定健康卡和院内 ID 关系接口

接口地址：`https://p-healthopen.tengmed.com/rest/auth/HealthCard/HealthOpenPlatform/ISVOpenObj/bindCardRelation`

该接口用于绑定和修改健康卡二维码和院内 ID 的映射关系，**适用于** 新用户建卡流程 、跨院用户一键注册健康卡，**请求参数为院内患者 ID 和二维码文本数据，响应参数为绑定结果。**

请求参数 req 说明：

参数名称	参数代码	必选	类型	说明
院内 ID	patid	是	string	院内 HIS 主索引
二维码文本	qrCodeText	是	string	

输入参数示例：

```
{
    "commonIn": {
        "appToken": "123",
        "requestId": "123",
        "hospitalId": "00000",
        "timestamp": "1525392000",
        "sign": "Q5vp1tdaHjuQpDK8yuDOAzFKTOQs5PxgzhLbxMpnadE="
    },
    "req": {
        "patid": "10086",
        "qrCodeText": "xxxx"
    }
}
```


响应参数 rsp 说明：

参数名称	参数代码	必选	类型	说明
绑定结果	result	是	boolen	true-成功，false-失败

输出参数示例：


```
{
    "commonOut": {
        "requestId": "123",
        "resultCode": 0,
        "errMsg": "成功"
    },
    "rsp": {
        "result": true
    }
}
```


3.2.7 用卡数据监测接口

接口地址：`https://p-healthopen.tengmed.com/rest/auth/HealthCard/HealthOpenPlatform/ISVOpenObj/reportHISData`

该接口用于接收医院上报的健康卡使用数据，**适用于** 用卡数据监测流程 **，请求参数为用户信息和健康卡使用场景，响应参数为** none。

请求参数 req 说明：

参数名称	参数代码	必选	类型	说明
二维码文本	qrCodeText	条件	string	如果证件类型**是健康卡**，则**二维码数据是必填**，反之则二维码数据非必填
身份证号码	idCardNumber	条件	string	如果证件类型**不是健康卡**，则**身份证号码必填**，反之则身份证号码非必填
姓名	name	条件	string	如果证件类型**不是健康卡**，则**姓名必填**，反之则姓名非必填
时间	time	是	string	格式:yyyy-MM-dd HH:mm:ss
医院ID	hospitalId	是	string	由医院扫码授权后生成，操作指引详见【邀请医院入驻】如何邀请医院入驻，授权绑定？
用卡环节	scene	是	string	参考用卡环节表
用卡科室	department	条件	string	如果用卡环节代码为 010101(挂号)、0101011(预约挂号)、0101012(当日挂号)，则用卡科室数据为必填。参考标准科室表
证件类型	cardType	是	string	01-居民身份证，其他参考证件类型表
用卡渠道	cardChannel	是	string	服务号、app、窗口等

输入参数示例：


```
{
    "commonIn":{
       "appToken": "123",
        "requestId": "123",
        "hospitalId": "10023",
        "timestamp": "1525392000",
        "sign": "Q5vp1tdaHjuQpDK8yuDOAzFKTOQs5PxgzhLbxMpnadE="
    },
    "req":{
        "qrCodeText":"123:1",
        "idCard":"360782199001010101",
        "name":"xxx",
        "time":"2018-10-02 16:36:57",
        "hospitalId":"10023",
        "scene":"010101",
        "department":"内科",
```

```
        "cardType":"01",
        "cardChannel":"服务号"
    }
}
```


响应参数 rsp 说明：

参数名称	参数代码	必选	类型	说明
占位	none	是	string	占位

输出参数示例：


```
{
    "commonOut":{
        "errMsg":"成功",
        "requestId":"72c9531093d34e0e82de448f95729c38",
        "resultCode":0
    },
    "rsp":{
        "none":""
    }
}
```


3.2.8 获取卡包订单 ID 接口

接口地址：`https://p-healthopen.tengmed.com/rest/auth/HealthCard/HealthOpenPlatform/ISVOpenObj/getOrderIdByOutAppId`

该接口用于获取卡包订单 ID`orderId`，进而配置跳转卡包页面的 URL，适用于 【统一卡包流程】二、医院服务号（领卡服务号）接入流程；`orderId` 一经使用即刻失效，下次使用需要重新获取；请求参数为服务号开发商用户凭证和二维码文本数据，响应参数为 `orderId`。

请求参数 req 说明：

参数名称	参数代码	必选	类型	说明
服务号开发商用户凭证	appId	是	string	开放平台官网分配的 appId
二维码文本	qrCodeText	是	string	

输入参数示例：

```
{
    "commonIn":{
        "appToken":"123",
        "requestId":"123",
        "hospitalId":"00000",
        "timestamp":"1525392000",
        "sign":""
    },
    "req":{
        "appId":"snzebh3x8zl4a4hi9",
        "qrCodeText":"2A4A19745AC6D6475C2AA48290B513F818F056277858104C89979A39E49D1444:1
"
    }
}
```


响应参数 rsp 说明：

参数名称	参数代码	必选	类型	说明
订单 ID	orderId	是	string	orderId 一经使用即刻失效，下次使用需要重新获取

输出参数示例：


```
{
    "commonOut":{
        "requestId":"",
        "resultCode":0,
        "errMsg":"成功"
    },
    "rsp":{
        "orderId":"df1ddbd7e8ef4475a656a70bf87804b3"
    }
}
```


3.2.9 注册批量健康卡接口

接口地址：`https://p-healthopen.tengmed.com/rest/auth/HealthCard/HealthOpenPlatform/ISVOpenObj/registerBatchHealthCard`

该接口用于批量升级已在医院服务号注册过就诊卡的历史用户数据，适用于[老用户批量升级健康卡流程]中批量注册操作，**请求参数为用户信息数组，响应参数为注册成功的二维码文本数据、健康卡 ID 等。**

请求参数 req 说明：

参数名称	参数代码	必选	类型	说明
用户信息	healthCardItems	是	数组	
离线微信身份码	wechatCode	是	string	离线微信身份码为固定值，不需要跳转链接获取，请联系对接人员获取
姓名	name	是	string	
性别	gender	是	string	男、女
民族	nation	是	string	汉族、满族等（如果没有该字段，请填未知）
出生年月日	birthday	是	string	格式：yyyy-MM-dd
证件号码	idNumber	是	string	
证件类型	idType	是	string	01-居民身份证，其他参考证件类型表
地址	address	否	string	
联系方式 1	phone1	是	string	
联系方式 2	phone2	否	string	
院内 ID	patid	是	string	院内用户唯一标识
用户 openId	openId	是	string	微信服务号用户唯一标识
微信跳转链接	wechatUrl	是	string	模板消息的 URL，即服务号健康卡列表页面的 URL

输入参数示例：


```
{
    "commonIn":
    {
        "appToken": "de725bef39fa83758a66aa31df8aecd2",
        "requestId": "72c9531093d34e0e82de448f95729c38",
        "hospitalId": "90003",
        "timestamp": "1543297690",
        "sign": "wwGFngfejKiNdi22J+1Kb5DfozWOWovLkIRupPyzzIM="
    },
    "req":
    {
        "healthCardItems": [
        {
            "wechatCode": "13B8AC208CEDACE7DD1DF57963150664",
            "name": "张三",
            "gender": "男",
            "nation": "汉族",
            "birthday": "2008-08-08",
```

```
            "idNumber": "100023200808081234",
            "idType": "01",
            "address": "地址",
            "phone1": "1888888888",
            "phone2": "1777777777",
            "patid": "patid",
            "openId": "openId",
            "wechatUrl": "https://open.tengmed.com"
        },
        {
            "wechatCode": "13B8AC208CEDACE7DD1DF57963150664",
            "name": "张三",
            "gender": "男",
            "nation": "汉族",
            "birthday": "2008-08-08",
            "idNumber": "100023200808081234",
            "idType": "01",
            "address": "地址",
            "phone1": "1888888888",
            "phone2": "1777777777",
            "patid": "patid",
            "openId": "openId",
            "wechatUrl": "https://open.tengmed.com"
        },
        ……//更多用户信息
        ]
    }
}
```


响应参数 rsp 说明：

参数名称	参数代码	必选	类型	说明
项目	rspItems	是	数组	
二维码文本	qrCodeText	否	string	注册成功时有值，失败则为空字符串
证件号码	idNumber	是	string	返回请求参数 req 中的 idNumber
健康卡 ID	healthCardId	否	string	如果 qrCodeText 是动态，则 healthCardId 有值

输出参数示例：


```
{
    "commonOut":{
        "requestId":"72c9531093d34e0e82de448f95729c38",
```

```
        "resultCode":0,
        "errMsg":"成功"
    },
    "rsp":{
        "rspItems":[
            {
                "qrCodeText":"二维码文本数据",
                "idNumber":"证件号码",
                "healthCardId":"健康卡卡号"
            },
            {
                "qrCodeText":"二维码文本数据",
                "idNumber":"证件号码",
                "healthCardId":"健康卡卡号"
            },
            ……//更多响应

        ]
    }
}
```

3.2.10 获取动态二维码接口

接口地址：https://p-healthopen.tengmed.com/rest/auth/HealthCard/HealthOpenPlatform/ISVOpenObj/getDynamicQRCode

该接口用于获取动态二维码，**适用于新用户建卡流程、老用户批量升级健康卡流程中展示动态二维码的环节，请求参数为健康卡 ID 和证件数据，响应参数为动态二维码文本及图片。**

请求参数 req 说明：

参数名称	参数代码	必选	类型	说明
健康卡 ID	healthCardId	是	String	
证件类型	idType	是	String	
证件号码	idNumber	是	String	

输入参数示例：

```
{
    "commonIn":
    {
        "appToken": "ec5d3215d774d7a3cb188839fb1662ba",
```

```
        "hospitalId": "20010",
        "requestId": "MCAC05C76D134E3D9C342GAFCF10ODYX",
        "sign": "Q3N1JdQn2p1v7111mwvg4vEREwsyDbw17eXP3WCcZEA=",
        "timestamp": "1543480720"
    },
    "req":
    {
        "healthCardId": "D109C4C340995DFC13262A55699FBDAD8S564E8B65796FA1",
        "idNumber": "432901198810228888",
        "idType": "01"
    }
}
```


响应参数 rsp 说明：

参数名称	参数代码	必选	类型	说明
二维码文本	qrCodeText	是	string	
二维码图片	qrCodeImg	是	string	BASE64

输出参数示例：


```
{
    "commonOut":
    {
        "errMsg": "成功",
        "requestId": "MCAC05C76D134E3D9C342GAFCF10ODYX",
        "resultCode": 0
    },
    "rsp":
    {
        "qrCodeImg": "二维码图片(BASE64)",
        "qrCodeText": "D109C4C340995DFC13262A55699FBDAD8S564E8B65796FA1:0:B0E3CC5F50EF5D
C663F094C985B58312::35020001001"
    }
}
```


3.2.11 动态二维码校验接口

接口地址：
`https://thealthopen.tengmed.com/rest/auth/HealthCard/HealthOpenPlatform/ISVOpenObj/verifyQRCode`

该接口用于校验动态二维码，**同时统计健康卡使用数据，适用于【扫码就医流程(可选)】2、动态二维码就医流程，请求参数为健康卡使用数据，响应参数为健康卡信息。**

请求参数 req 说明：

参数名称	参数代码	必选	类型	说明
二维码文本	qrCodeText	是	String	
终端 Id	terminalId	是	String	
时间	time	是	String	格式 yyyy-MM-dd HH:mm:ss
用卡环节	medicalStep	是	String	参考用卡环节表
用卡城市代码	useCityCode	是	String	
用卡城市名称	useCityName	是	String	
医院代码	hospitalCode	是	String	
医院名称	hospitalName	是	String	
发卡机构	orgId	否	String	
姓名	name	否	String	
身份证号码	idCard	否	String	
使用场景	useScene	否	String	参考用卡环节表
证件类型	useType	否	String	01-居民身份证，其他参考证件类型表
用卡渠道	useChannel	否	String	参考用卡渠道表

输入参数示例：


```
{
    "commonIn":
    {
        "appToken": "ec5d3215d774d7a3cb188839fb1662ba",
        "requestId": "MCAC05D76AEA4E3AA23A23AACG10DAYZ",
        "hospitalId": "20010",
        "timestamp": "1543480753",
        "sign": "H1B5oQvHmKXTm5xcfFOdBgBojBQdVtwcsNTCv558b5w="
    },
    "req":
    {
        "qrCodeText": "D109C4C340995DFC13262A55699FBDAD8S564E8B65796FA1:0:B0E3CC5F50EF5D
C663F094C985B58312::35020001001",
        "terminalId": "a12111134123",
        "time": "2018-11-29 16:10:33",
        "medicalStep": "010101",
        "channelCode": "01",
        "channelName": "人工窗口",
        "useCityCode": "110000",
        "useCityName": "北京市",
        "hospitalCode": "01",
```

```
        "hospitalName": "人民医院",
        "orgId": "发卡机构",
        "name": "姓名",
        "idCard": "身份证号码",
        "useScene": "使用场景",
        "useType": "证件类型",
        "useChannel": "用卡渠道"
    }
}
```


响应参数 rsp 说明：

参数名称	参数代码	必选	类型	说明
地址	address	是	string	
出生日期	birthday	是	string	格式 yyyy-MM-dd
健康卡 ID	healthCardId	是	string	
证件号码	idNumber	是	string	
证件类型	idType	是	string	
健康卡主索引	phid	是	string	
姓名	name	是	string	
民族	nation	是	string	
电话号码	phone	是	string	
性别	gender	是	string	

输出参数示例：


```
{
    "commonOut":
    {
        "errMsg": "成功",
        "requestId": "MCAC05D76AEA4E3AA23A23AACG10DAYZ",
        "resultCode": 0
    },
    "rsp":
    {
        "address": "未知地址",
        "birthday": "1988-10-22",
        "gender": "女",
        "healthCardId": "D109C4C340995DFC13262A55699FBDAD8S564E8B65796FA1",
        "idNumber": "432901198810228888",
```

```
        "idType": "0",
        "name": "龙小英",
        "nation": "",
        "phid": "B146D0EA2BB828FC598D64761305A482F955F670193AD7CAA8790DC422463AB7",
        "phone": "18888882012"
    }
}
```


3.2.12 全局返回码

错误码	错误内容	错误说明
0	请求成功	
1	适配平台出错	联系工作人员协助排查
-1	卡管平台异常	联系工作人员协助排查
-2	HTTP 服务异常	联系工作人员协助排查
10001	发生了异常	联系工作人员协助排查
10002	请求参数 req 不能为空	
10003	请求参数 commonIn 不能为空	
10004	uid 不能为空	
10005	数据库错误	联系工作人员协助排查
10006	健康卡号不能为空	
10007	健康卡授权码不能为空	
10008	健康卡授权码不存在	可能是已使用或已失效
10009	健康卡授权码不合法	
10010	系统接口异常，请尝试刷新页面	微信身份码 wechatCode 不存在，可能是已使用或已失效
10011	微信身份码不合法	
10012	微信身份码不能为空	
10013	姓名不能为空	
10014	证件号不能为空	
10015	核身不通过	
-10017	获取健康卡失败	
10018	二维码不能为空	
-10019	健康卡不存在	
10020	二维码识别失败	
10021	健康卡授权码检验异常	

-10022	该手机号码对应的电子卡已经存在，请勿重复申请	
-10023	该证件号码对应的电子卡已经存在，请勿重复申请	
-10024	终端信息不合法，调用失败	
-10025	缺少参数（姓名、性别、民族、手机号码、证件类型、证件号码为必填项）	
-10026	身份证格式不合法	
-10027	身份证不合法	
-10028	主索引合适身份证号码重复	
-10029	请检查加密机链接	
10030	没有电子健康卡号	
-10031	该电子卡已被冻结或注销，无法绑定	
-10032	该身份证号码对应的电子卡已经存在，请勿重复申请	
10033	患者 ID 不能为空	
10034	接口调用次数限制	扫码入驻七天后仍未上线的医院，**限制接口调用次数**，具体为 100 次/天/接口，上线后自动解除限制。
10035	业务流水号不能为空	
10036	消息推送时间不能为空或格式错误	
10037	requestId 字段长度超出限制（512byte）	
-10038	核身异常	
-10039	ocr 识别失败	
-10040	ocr 失败异常	
10041	ocr 图片内容不得为空	
10042	参数错误	
10043	手机号不得为空	
-10044	卡管平台返回错误	联系工作人员协助排查
10048	更新健康卡信息失败	
10050	开始日期不能大于或等于结束日期	
10051	日期格式错误	
10053	订单号不存在	
10054	重复请求	
10055	模板 ID 不得为空	

10056	输入的姓名与卡管平台保存的姓名不一致	
10057	绑定人数已达上限，请前往电子居民健康卡小程序解绑	
-110086	卡管平台返回错误	
20000	鉴权失败	
20001	参数格式错误	
20002	无效参数	
20003	业务服务响应格式错误	
20004	请求参数过长	
21001	appToken 不能为空	
21002	非法 appId 或 appSecret	
21003	appToken 超时	
21004	非法 appToken 或 hospitalId	
21005	无权操作该医院 ID	
21006	签名错误	
21007	接口调用被限制	

3.3 附录

3.3.1 用卡环节表

用卡环节代码	名称	说明
010101	挂号	成功状态
0101011	预约挂号	
0101012	当日挂号	
0101013	挂号记录	
0101014	预约导诊	
0101015	当日取号	
010102	诊断	
0101021	排队候诊	
0101022	门诊记录	
0101023	病历打印	
010103	取药	
0101031	取药记录	
010104	检查	成功状态

0101041	预约检查	
0101042	预约检查记录	
010105	收费	成功状态
0101051	门诊缴费	
0101052	门诊缴费记录	
0101053	住院缴费	
0101054	住院缴费记录	
0101055	门诊充值	
0101056	住院充值	
010106	开方	成功状态
010108	取（查询）报告	成功状态
0101081	取（查询）检查报告	
0101082	取（查询）检验报告	
0101083	取（查询）体检报告	
010109	住院登记	
00000	其他	成功状态
00001	便民服务	
00002	孕期保健	
00003	儿童保健	
00004	家庭医生	
00005	在线问诊	
00006	计划免疫	
00007	个人信息	

3.3.2 证件类型表

证件类型	证件名称
01	居民身份证
02	居民户口簿
03	护照
04	军官证
05	驾驶证
06	港澳居民来往内地通行证
07	台湾居民来往内地通行证
08	出生医学证明

09	医保卡
10	就诊卡
11	健康卡
99	其他法定有效证件

3.3.3 用卡渠道表

用卡渠道代码	用卡渠道类型名称	说明
0100	人工窗口	
0200	自助机	
0300	医生工作台	
0400	微信渠道	
0401	服务号	
0402	小程序	
0000	其他	APP、支付宝等

3.3.4 用卡费别表

费别代码	**费别名称**	**说明**
0100	自费	
0200	医保	
0300	公费	
0000	其他	

3.4 FAQ

1. 请求参数和请求数据可否随意构造？

不可以，需按照接口规范构造参数和数据。

2. requestId 有什么作用，可否使用同一个？

不可以，requestId 是为了防止重放攻击和排查问题，建议每次请求生成不同的 requestId

3. 跳转页面是否支持 ip+端口的形式？

支持，用作**域名白名单**配置

4. 接口请求的编码格式有要求吗？

需使用 utf8 的编码格式，注意要在 header 里面进行设置

5. 同一个身份证，重复建卡，二维码会变更吗？

二维码文本**不会变更**；同时，如果重复建卡提供的手机号等信息有差异，也不会进行更新，以**第一次注册**为准；